End-to-end Predictive Models for Remote Sensing Applications

John Ray Bergado

END-TO-END PREDICTIVE MODELS FOR REMOTE SENSING APPLICATIONS

D I S S E R T A T I O N

to obtain
the degree of doctor at the University of Twente,
on the authority of the rector magnificus,
prof.dr.ir. A. Veldkamp,
on account of the decision of the Doctorate Board,
to be publicly defended
on Thursday, December 17, 2020 at 10.45 hrs

by

John Ray Bergado
born on July 7, 1992
in Morong, Bataan, Philippines

This dissertation is approved by:

prof.dr.ir. A. Stein (promoter)
dr. C. Persello (co-supervisor)

ITC dissertation number 389
ITC, P.O. Box 217, 7500 AE Enschede, The Netherlands

ISBN: 978-90-365-5096-3
DOI: http://dx.d oi.org/10.3990/1.9789036550963
Printed by: ITC Printing Department, Enschede, The Netherlands

UNIVERSITY OF TWENTE.

FACULTY OF GEO-INFORMATION SCIENCE AND EARTH OBSERVATION

Graduation committee

Chair

prof.dr. F.D. van der Meer	

Supervisor

prof.dr.ir. A. Stein	University of Twente / ITC

Co-supervisor

dr. C. Persello	University of Twente / ITC

Members

prof.dr. R. Zurita Milla	University of Twente / ITC
prof.dr.ir. R.N.J. Veldhuis	University of Twente / DMB
prof. E. Pebesma	WWU Münster, Germany
prof.dr. B. Demir	TU Berlin, Germany

Acknowledgements

Thanks be to God for surrounding me with an amazing and ever supportive group of people—my supervisors, colleagues, family, and friends—who have helped me, every step of the way, throughout my entire Ph.D. journey.

I would like to express my most sincere gratitude to my daily supervisor, Claudio Persello, for his persistent dedication and ingenuity—convincingly guiding me to become a better researcher, not settling for anything less than excellent, but also bringing my feet back to the ground when certain ideals paralyze the progress of my research. His great expertise in machine learning stimulated the technical development of all my ideas in this thesis. Also, my deepest thanks go to my promoter, professor Alfred Stein, for consistently helping me to frame the context of my research. His mastery of spatial statistics and immense experience over different application domains were invaluable to helping me highlight the contributions of my research.

I would also like to thank everyone whom I met in UT and RMIT. Marc Demange for accommodating me during my secondment in RMIT; Karin Reinke for sharing her expertise in remote sensing of bushfires; Teresa, John, Roelof, Loes, Karen, Petra, and Tonny for helping me with all the administrative tasks in UT; and the committee members for dedicating time and effort to read and review this manuscript.

Also, I would like to acknowledge my officemates, Dewi, Vera, Sarah, Anurag, and Wufan for keeping the office a peaceful and stimulating place to work in. Special thanks to all friendly colleagues I met outside of our department: Matthew, Oliver, Riswan, Tang, and Sonia for the travels and dinners we have shared together. To all the friends who've been there during my Ph.D.: Beverly, Celeste, Kate, Jen, Nick, Fang, Ipsit, Edson, Joshua, Rita, RJ, and Jam, thank you for all the memorable experiences we had, helping me keep my composure when I am struggling with my research woes.

I am grateful to my best friend and the love of my life, Bhuwan, for her

unwavering support.

I thank my family, my father, Bernie, my mother, Neth, my brother, Imbo, and my two sisters, Ame and Yeye, for always having my back in every aspect of life.

Finally, I would like to thank the European Commission (funding institution of the GEOSAFE project, granted under the European Union's Horizon 2020 research and innovation programme, Marie Skłodowska-Curie grant, agreement No 691161) for the opportunity I had to spend part of my Ph.D. in RMIT, Melbourne—meeting people with deep expertise in the science of bushfire.

Contents

List of Figures

List of Tables

Introduction

1

1.1 Learning to Predict

A prediction is a guess about an uncertain but realizable phenomenon. The guess can be based on our prior experience, beliefs, knowledge, and understanding of the phenomenon; or more objectively, be based on past observations of possible factors causing the phenomenon. The phenomenon itself can be an inferable fact (e.g. the greenish pixels in the image are trees) or a contingent event (e.g. the time and location of a likely wildfire ignition). Below we discuss a quantitative perspective of prediction in the context of remote sensing applications.

1.1.1 Function Approximation as a Form of Predictive Modeling

Predicting, i.e. making predictions, plays a crucial role in several remote sensing applications ranging from land cover mapping [24, 156] to wildife risk management [14, 62]. Often, we present predictions in quantitative form: for example, i) how much area of the city is still covered by green vegetation or ii) how much area is likely in risk due to a nearby wildfire. In this work, we refer to predictive models as formalizations of methods to make and quantify these predictions. One simple way to formalize such models is *function approximation* [58, pp. 28–32]—where we try to find a useful approximate function $\hat{f}$ for the true underlying relationship

$$\mathrm{y} = f(\mathbf{x}) \tag{1.1}$$

between the input vector $\mathbf{x}$ and output y. In statistics, $\mathbf{x}$'s are interchangeably called independent variables, *predictors*, or *covariates*; while the term *features* is often preferred in machine learning and pattern recognition. y's (classically termed dependent variable), on the other hand, are interchangeably called

response or *target variables*. Following our land cover mapping example: $\mathbf{x}$ is a vector of b predictors ($\mathbf{x} = \{\mathrm{x}_1, \mathrm{x}_2, ..., \mathrm{x}_b\}$) that can naively correspond to the bands of the remotely sensed image being classified, e.g. realizations of x_1 are the pixel values in the near infrared band; and y is a scalar representing c land cover classes ($\mathrm{y} = \{\mathrm{y}_1, \mathrm{y}_2, ..., \mathrm{y}_c\}$), e.g. realizations of y_1 are the pixels in the classified map corresponding to grassland cover.

We can then view other predictive modeling tasks $\mathtt{t}$ in similar remote sensing applications as a mapping of the predictors to the response variables via an approximate function $\hat{f}$. Intuitively, for different kind of tasks, data, and objectives—we need to approximate $\hat{f}$ in a different way. We expound the ways of approximating a function for predictive modeling in the following paragraphs. Table 1.1 provides a simplified overview of these different approaches.

Table 1.1 Summary of approaches in predictive modeling

Property of Modeling Task $\mathtt{t}$	Different Types
Availability of labels in y	Supervised vs. Unsupervised
Level of measurement in y	Classification vs. (Metric) Regression
Form of y	Structured vs. Scalar Prediction
End-goal of $\mathtt{t}$	Data-driven vs. Knowledge-driven
Representation of $\mathbf{x}$	Feature Learning vs. Engineering

1.1.2 Supervised vs. Unsupervised

There are generally two ways to determine a useful $\hat{f}$ for a given task $\mathtt{t}$. We can opt to approximate $\hat{f}$ either in a supervised or unsupervised manner. If we collect target or reference observations for y, we can learn $\hat{f}$ in a supervised manner. But if target observations are unavailable for y, we need to learn $\hat{f}$ in an unsupervised manner.

In the supervised case, the available observations in y serve as a teaching/supervising signal (hence the term supervised) that directs the model to learn a good $\hat{f}$. The "supervisor" or "teacher" (roughly corresponding to the observations in y) either associates a correct answer and/or an error to the "student's answer" (which is $\hat{\mathrm{y}}$, our current model's prediction)—much like a classroom metaphor [58, pp. 485–487]. Hence, in most supervised learning algorithms, we update our $\hat{f}$ based on the feedback we obtain from comparing $\hat{\mathrm{y}}$ against our assumed truth: the observations we have for y. Building upon our land cover mapping application, the observations in y may come from image

interpretation, ground surveys, secondary sources such as topographic maps, etc. [30, pp. 85–103]; [117, pp. 296–297].

In the absence of available observations in y, we resort to unsupervised methods. Most machine learning literature does not associate unsupervised learning methods to prediction problems, but rather refer to other tasks such as dimensionality reduction and structure discovery [107, pp. 9–16]; [70, pp. 373–374]. This absence of responses in y makes the end-target of unsupervised problems implicitly defined or undefined. Despite this lack of explicit definition of y, we are still learning to model our input $\mathbf{x}$ into some other form, e.g. representations of $\mathbf{x}$ with reduced dimension or clusters discovered within $\mathbf{x}$. Hence, we can still arguably view unsupervised learning problems in a similar perspective we developed for predictive modeling: approximating a proper $\hat{f}$ for implicitly defined or undefined y's. Furthermore, remote sensing applications use clustering methods (such as *ISODATA* [8] and *K-means* [94]) to perform unsupervised predictions. Following our land cover mapping example, a remotely sensed image may be classified into unknown land cover classes using an unsupervised (clustering) method. These unknown classes will then be labeled by a human operator [117, p. 249].

1.1.3 Classification vs. Regression

For better clarity in terminologies, we further distinguish forms of predictive modeling into two types depending on the level or scale of measurement in y. We perform classification when the response variables are categorical, either nominal (without order) or ordinal (with order); and we perform regression when the response variables are continuous. The choice of appropriate form inherently depends on the nature of the predictive modeling task, e.g. classification for mapping discrete objects while regression for mapping continuous surface representations. Mapping land cover may fall under a classification problem while mapping a continuous wildfire risk index may be treated as a regression problem.

Classification drives many remote sensing applications such as land cover mapping, flood mapping [125], mapping of soil and minerals [104], etc. In this era of multisource, voluminous, online-stream of data (see the 3 dimension of "difficult" also called "big" data [79])—automating the classification of remotely sensed data becomes relevant especially for environmental monitoring systems [39] integrating remote sensing technology. The authors in [88] and [117, pp. 193–266] review and discuss a number of well-known automated image classification methods. Some of these methods include the widely-used *maximum likelihood classifier* (MLC) [117, pp. 194–204], and machine learning

methods like *support vector machines* (SVM) [117, pp. 226–231] and *artificial neural networks* (ANN) [117, pp. 232–242]. Several points of concern affect the automation of these classification methods such as appropriate choice and representation of data [88], and assessment of the classification accuracy [43]. We will further deal with their relevance, specifically data representation, in 1.1.6.

Other remote sensing applications such as estimation of environmental factors (e.g. vegetation health, pollutant concentration, etc.) benefit from regression. We generally represent these environmental factors in terms of continuous indices—hence, we perform regression instead of classification. In [41], the authors review the application of linear regression for predicting biophysical factors such as the leaf area index (LAI) and simple ratio (SR) vegetation index. Some more advanced methods like *Gaussian process regression* (GPR) [112] and *support vector regression* (SVR) [142] were also applied to similar estimation of biophysical factors from remotely sensed images.

1.1.4 Structured Prediction vs. Scalar Prediction

Aside from the level of measurement, the form or dimensionality of y also distinguishes two approaches in predictive modeling. If y is zero-dimensional, we perform scalar prediction; but as soon as we add dimension/s to y and consider the relationships between the elements of the latter, we will refer to the approach as structured (output) prediction. In our land cover mapping example, scalar prediction is equivalent to predicting a class for an individual pixel at a time. On the other hand, we will be performing structured prediction if we predict land cover classes for multiple pixels—which, unlike object-based classification, may contain pixels with different classes.

Intuitively, the additional dimension and organization of output makes structured prediction problems more complex than their scalar prediction counterparts. Hence, most work in the context of remote sensing are performing scalar prediction. But some recent works [2, 145], specifically in the context of remotely sensed image analysis, steer into the direction of structured learning. Both uses *conditional random fields* (CRF) [78] to model the structure in their output predictions.

1.1.5 Data-driven vs. Knowledge-driven

We can distinguish two kinds of predictive models: knowledge-driven and data-driven models. Here, we classify knowledge-driven models as those heavily relying on inputs derived from the upper strata of the data-information-

knowledge-wisdom (DIKW) pyramid [119]. The distinction between each strata can arguably be subjective. In a more concrete analogy, data-driven is to supervised learning methods as knowledge-driven is to rule-based methods. In rule-based methods, the user provides the rules embedded in $\hat{f}$; while in supervised learning methods, the user provides examples and the algorithm learns the appropriate input to output mapping based on the examples given to it. We can further loosely generalize the difference between the two as being a trade-off between predictive accuracy and model interpretability [73]—where, in some applications, one may be favored than the other. In general, knowledge-driven models are more interpretable than their data-driven counterparts; and data-driven models demonstrate higher predictive accuracy than their knowledge-driven counterparts.

Before the surge in abundance of remotely sensed data [92], insufficient computing power and data scarcity primarily limits the choice of predictive modeling approach. Most remote sensing application then resort to knowledge-driven models, like a rule-based method such as [121]. Since constructing such knowledge-driven models requires much less resources—in terms of data and computing power—as compared to data-driven models, like the use cases illustrated in [80] employing machine learning methods. But all the advances in sensor, data acquisition, and computing technologies paved the way to use more complicated and resource-intensive models. In the end, the choice of which kind of approach to use will largely depend on the end-goal of the modeling task: either we value predictive accuracy over model interpretability (hence preferring data-driven models) or vice-versa.

1.1.6 Feature Learning vs. Feature Engineering

Lastly, we distinguish how we represent our features $\mathbf{x}$. To avoid confusion, we differentiate our input data $\mathbf{x}$ from our features $\mathbf{r}$

$$\mathbf{r} = \phi(\mathbf{x}) \tag{1.2}$$

such that they are related by the function ϕ—mapping the input data into the feature space. A good choice of ϕ can greatly improve the predictive performance of our model. We can either learn ϕ directly from the input data available to us or we can construct it based on our knowledge of the predictive modeling task. We call the first approach feature learning or representation learning [49, pp. 12–15]; [50, pp. 4–5]. We call the second one feature engineering or feature handcrafting.

This last distinction in predictive modeling approach is directly related to the two previous approaches. Feature learning is more data-driven than

knowledge-driven, while feature engineering is more knowledge-driven than data-driven. Hence, the same illustration of our land cover mapping example applies. We can either manually construct ϕ depending on our knowledge base of the problem or we can plug-in ϕ to our function approximation algorithm. Following the latter approach, Equation 1.1 becomes

$$\mathrm{y} = f(\phi(\mathbf{x})) \tag{1.3}$$

emphasizing the difference between our input data $\mathbf{x}$, features $\mathbf{r}$, and feature mapping function ϕ. Deep learning is a specific case of feature learning where models are composed of multiple processing layers gradually transforming the input data into a proper feature representation $\mathbf{r}$ and finally extracting f:

$$\mathbf{r} = \phi_n(\phi_{\dots}(\phi_1(\phi_0(\mathbf{x})))) \tag{1.4}$$

with n composite functions transforming $\mathbf{x}$ into $\mathbf{r}$.

1.2 Remote Sensing Applications

Predictive modeling can be useful for several remote sensing applications [24, 156, 14, 62]. For this research, we are particularly interested in three applications: land cover classification, land use classification, and wildfire risk prediction. Input data can come from a wide range of sensors varying in the spatial, spectral, and temporal resolution. Deep learning allows us to build predictive models in an *end-to-end* manner, generating predictions directly from the input data by integrating conventionally separate processing steps, e.g. manual feature extraction, within the model itself.

1.2.1 Land Cover and Land Use Classification

Urbanization continues to change the anthropogenic landscape [35]. We need efficient methods to automatically update land cover maps—a data product essential for environmental authorities and policy makers to carry out well-informed decisions. For local applications—where objects of interests such as roads, individual buildings, trees, etc. must be mapped—very high resolution images from airborne or satellite platforms may be required to derive necessary land cover maps. Not only does finer resolution images suit these kind of applications, such images also reduces the effect of the mixed pixel problem [89]. But with higher spatial resolution comes higher spectral intra-class variation that may cause difficulty in the classification problem.

Spatial-contextual classifiers [83] address the spectral variation problem by taking into account the information around a group or neighborhood of pixels. Such classifiers use handcrafted features—e.g. texture from the gray level co-occurrence matrix (GLCM) [57], local binary patterns (LBP) [109], etc.—to extract spatial-contextual information. But optimizing the proper configuration of these feature extraction methods can be inefficient and time consuming—especially for very high resolution images where long distance pixel-to-pixel dependency is expected. Aside from using handcrafted features, other classification approaches on very high resolution images even try to model the mapping of these features to corresponding class labels using handcrafted classification rules [163]. Classifiers following such methods fall under the knowledge-driven and feature engineering type of approach (see 1.1.6 and 1.1.5). A more data-driven approach is to learn the features and their corresponding mapping to classification labels directly from the data (see Figure 1.1).

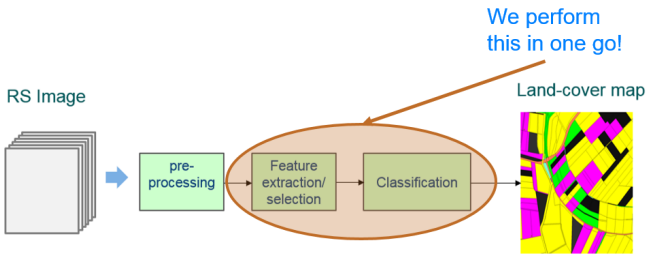


Figure 1.1 Learning features and classification rules simultaneously from the data.

1.2.2 Land Use Classification

Land use is another vital information for various planning and policy-making processes [64]. Land use describes the human activity attached to a specific geographical location. For example: buildings used for either residential or commercial purposes, open spaces for recreation or waste management, trees for timber supply or for natural reserves, etc. Intuitively, automatically classifying land use compared to land cover will be more difficult since land use classes are defined in a higher abstraction (and finer-grained) level. Hence, limited work has been done in pixel-wise classification of land use from remotely sensed images.

In [134], the authors combined vegetation indices—normalized difference vegetation index (NDVI) and transformed difference vegetation index (TDVI),

textural measures from gray-level co-occurrence matrix (GLCM), and edge density to classify land use from IKONOS imagery of an Italian region using maximum likelihood classification. The authors in [162] employed a rule-based method to classify urban land use from another IKONOS scene of Ontario Canada. Other works such as [91] and [25] inferred land use from LIDAR derived features and single polarized SAR data respectively. All of the works mentioned performed land use classification by engineering features by hand, with even [162] manually specifying the classification rules to be applied.

1.2.3 Wildfire Prediction

Wildfire continues to be one of the major environmental problems in the world [154]. To help land and fire management agencies manage and mitigate wildfire-related risks, we need to develop tools for mapping the hazards and risks associated with wildfire. Remote sensing coupled with ancillary data, such as ground-based sensor observations and topographical datasets, can help us characterize the dynamics of wildfire related events [28]. One such characterization is the quantification of the probability of a wildfire burn.

Estimates of the probability of wildfire burn can either directly serve as a proxy measure for wildfire risk or may serve as input, together with information about assets-at-risk and their corresponding vulnerabilities, to probabilistic methods from the actuarial sciences to quantify wildfire risk [42]. Furthermore, this probability can also guide problems on wildfire response and fuel management, e.g. probability of burning as an input to prescribed burning optimization.

Producing this probability out of input variables extracted from a heterogeneous stack of data including time-series of remotely sensed images, meteorological observations, and geospatial layers from topographical databases can be complicated. Most studies employ a logistic regression (LR) [28, 5] trained on a number of relevant wildfire indicators and information on historical wildfire locations. The single-level linear combination employed in an LR limits the complexity of the function mapping input variables to the probability of wildfire burn. Higher-level spatial and temporal association (intermediate feature representations) between the input variables may improve the estimates of the probabilities. Just like the mapping function, however, there is a knowledge gap on how to construct these higher-level features.

1.3 Research Problem

Relevant remote sensing applications like urban land cover and land use classification and mapping probability of wildfire events require the organization and analysis of challenging geodata. Challenging in terms of dealing with large volume (very high resolution satellite and aerial imageries), high velocity (weekly, daily, and subdaily time series of remotely sensed observations), highly heterogeneous (varying data structure, quality, and storage format) datasets. Current representations of the input data may be insufficient to effectively perform a prediction task related to the problem. We need to learn further representations of our input data that can improve the predictive performance of our models.

The main focus of this research is the use of deep learning methods on these challenging remote sensing applications. Learning higher-level data representations is central to deep learning. One can find several formulation of the definition of deep learning [34]. For the sake of clarity, we adopt a modified version of the definition presented in [52]: deep learning is a group of techniques facilitating the learning, retrieval, and analysis of information (higher-level data patterns and representations) that are deeply hidden in the input data. These techniques perform the learning of representations in a hierarchical and distributed manner. Hierarchical, in a way that deeper (higher-level) representations are built on top of simpler (lower-level) ones; and distributed, in a way that inputs are described by multiple features and each feature participates in the representation of multiple inputs [50, pp. 13–19]. Deep learning not only optimizes the features learned on for the prediction task, but also streamlines and objectifies the prediction processing pipeline—skipping tedious and subjective feature engineering steps.

1.4 Research Objectives

This research aims to develop deep learning methods for building end-to-end predictive models in remote sensing. Variants of deep neural networks will be mainly employed for three applications: land cover classification, land use classification, and wildfire prediction. We formulate the work into four key objectives:

1. To develop a deep learning based method performing an end-to-end image fusion and classification of a multiresolution VHR satellite image in the context of urban land cover classification.

2. To develop a deep learning based method to model contextual *label-to-label* dependencies and effectively regularize classification maps in the context of urban land cover classification.
3. To develop a deep learning based method to classify urban land use from VHR satellite images.
4. To develop a deep learning based method predicting daily maps of the probability of a wildfire burn.

1.5 Thesis Outline

Chapter 1 presents background information on predictive modeling relating it to several concepts in machine learning and remote sensing applications, the latter being employed as use cases in this thesis. The chapter also presents the research problem, corresponding research objectives, and the outline of this thesis.

Chapter 2 provides an overview of several deep learning concepts used in this study.

Chapter 3 presents a multiresolution convolutional network for urban land cover classification. The network embeds both image fusion, feature extraction, and image classification in a single end-to-end framework.

Chapter 4 presents a recurrent convolutional network to model contextual label-to-label dependencies and effectively regularize urban land cover classification maps. Contextual label dependencies are incorporated in the recurrent convolutional network by feeding classification scores of a previous convolutional network instance to a succeeding one.

Chapter 5 presents a deep fully convolutional multitask network to perform urban land use classification from VHR imagery. Urban land cover classification is used as a complimentary task in training the multitask networks.

Chapter 6 presents a fully convolutional network for predicting daily maps of the probability of a wildfire burn over the next 7 days for Victoria, Australia over the period of 2006–2017. The network utilizes as an input an extensive set of wildfire related variables taken from various data sources such as: time series of satellite images and data products, climatological sensor observations, topographical geospatial databases, and historical wildfire burn records.

Chapter 7 presents the synthesis of this thesis. Key results from the previous chapters are summarized. The chapter ends with the conclusions and recommendations based on these summarized findings.

End-to-end Predictive Models

2

Deep learning presents a promising way to build end-to-end computational models by learning hierarchical and distributed representations of data. Hierarchical, in a way that deeper (higher-level) representations are built on top of simpler (lower-level) ones; and distributed, in a way that inputs are described by multiple features and each feature participates in the representation of multiple inputs [50, pp. 13–19]. It provides a framework where the transformations to construct the feature representations and the rules for predictions are learned simultaneously, integrating conventionally independent pre- and post-prediction steps, and delivering end-to-end predictive models. It stands on the premise that some mapping functions may be more efficiently approximated by deeper architectures compared to their shallower counterparts [11]. This framework results in highly flexible models that have empirically shown outstanding improvement of state-of-the-art methods in several applications. See [127] for an exhaustive review of benchmark results relevant to deep learning.

More specifically, we are interested in applying deep learning using a family of models called artificial neural networks. Variety of architectures exists such as convolutional neural networks (CNN), recurrent neural networks (RNN), autoencoders, Boltzmann machine variants, etc.—each of which are generally tailored to certain applications. For example, using CNN for images and using RNN for sequential data.

2.1 Artificial Neural Networks

Artificial neural networks are a group of statistical learning models inspired by the structure of biological brains of animals. Computational units called artificial neurons (perceptrons) comprises these networks. We characterize a neural network by: i) how the artificial neurons are organized, ii) the operation each artificial neuron performs, and iii) the learning rules governing them [36, pp.

9–12]. Synapses of the biological brain exhibits some form of plasticity (changing or vanishing of connection strength) that is suggested to drive the underlying process on how memory and learning works [36, pp. 1–5]. Analogous to the biological synapses, the weights (or parameters) of the connections between the units of a network also changes as the network is trained to learn a specific task, e.g. image classification. The operation performed by an artificial neuron can be summarized by an affine transformation (sum of products of the weight of the connections and the value of the preceding units)

$$\mathrm{a_j} = \mathrm{w_{0,j}} + \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathbf{z}_{\mathrm{i,j-1}} \mathbf{W}_{\mathrm{i,j}} \tag{2.1}$$

followed by a non-linear transformation. For example, the sigmoid

$$\mathrm{z_j} = \frac{1}{1 + \exp(-\mathrm{a_j})} \tag{2.2}$$

or the hyperbolic tangent

$$\mathrm{z_j} = \tanh(\mathrm{a_j}) \tag{2.3}$$

functions; where $\mathrm{a_j}$ is a pre-synaptic activation of a neuron in the j^{th} layer with n connections from the preceding $\mathrm{j}-1^{th}$ layer, $\mathrm{z_j}$ is the post-synaptic activation, $\mathrm{w_{0,j}}$ is the weight of the bias unit, and $\mathbf{W}_{\mathrm{i,j}}$ is the matrix containing the weights of the connections. Figure 2.1 illustrates the operations performed by a single artificial neuron where: $\mathrm{x_1}, \mathrm{x_2}, ...\mathrm{x_n}$ are the units, having weights of $\mathrm{w_1}, \mathrm{w_2}, ...\mathrm{w_n}$, with incoming connection; $\mathrm{b_0}$ is the bias unit; "Σ" sums the product of the incoming units with their weights; and "$\sim$" is the non-linear operation applied (see Equations 2.2 and 2.3).

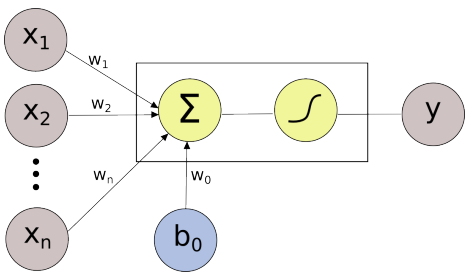


Figure 2.1 Diagram of an artificial neuron (also called perceptron) [12].

A typical example of an artificial neural network is a multilayer perceptron (MLP). In an MLP, we group the units into three kinds of layers: input, hidden, and output. Each unit connects to all other units in the preceding (except for those in the input layer) and succeeding layers while being disconnected to neurons in the same layer (see Figure 2.2). The units in the input layer takes a vectorized form of the data, e.g. in an image it will be the digital numbers of a pixel in each band. The units in the hidden layer are the feature representations of the data. Finally, the output layer contains the corresponding results of a classification or regression problem, e.g. the class label scores in a land cover classification. Aside from the characterization of the output units, the problem is expressed by formally defining a loss or objective function to be minimized. In a supervised learning setting (see 1.1.2), we often minimize this objective function using the backpropagation [120] with gradient descent algorithm. For more details regarding the MLP, the reader is directed to [36] and [50]. All the details mentioned here are also applicable to other varieties of artificial neural network architecture as we will discuss below.

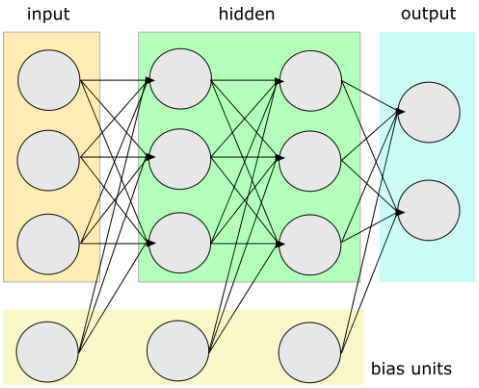


Figure 2.2 Simplified structure of an MLP [12].

MLP can be applied in a number of remote sensing applications [97] involving classification or regression problems. Specific examples of applications are land cover classification [123], land use classification [26], and wildfire ignition prediction [33]. Interestingly, [123] and [26] both used SAR data and all three examples employed MLP with no more than two hidden layers.

2.2 Convolutional Networks

Convolutional neural network (CNN) belongs to a group of artificial neural networks whose hidden layers employ convolutional and aggregational (pooling) operations. The CNN architecture originated from Fukushima's neocognitron [45], inspired by Hubel and Wiesel's hierarchical model of the visual cortex [66], with the main difference of the original CNN being trained via the backpropagation with gradient descent algorithm [120]. The mentioned hierarchical visual cortex model comprises of group of cells with "simple" and "complex" receptive fields. In analogy, the neocognitron network [45] comprises of hierarchical layers consisting of alternating S-cells and C-cells resembling the simple and complex cells in the visual cortex model. S-cells serve as feature extractors responding to specific signal patterns, while C-cells receive and subsample shifted versions of activation signals from a group of preceding S-cells—allowing a certain degree of tolerance in the change in position of the patterns [44]. Similarly, convolutional and pooling layers in the CNN performs the same pattern detection and subsampling with small positional shift invariance respectively. The filters in the convolutional layers encode the patterns learned by the network. By using filters with receptive fields, or filter sizes, smaller than the dimension of the input signal (e.g an image), the same filter may be used to recognize similar patterns in different locations of the input. This "filter reusing" (formally: parameter sharing) scheme of a convolutional layer results to a network with significantly lower number of parameters than an equivalent non-convolutional (fully-connected) version. See figure 2.3 to see the difference between a convolutional and non-convolutional layer.

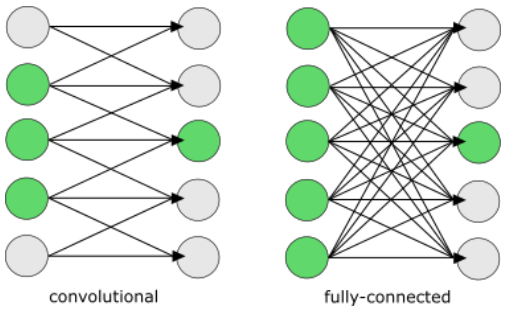


Figure 2.3 Difference between convolutional and fully-connected layers [12].

The pooling layer then performs a fixed operation (e.g. taking the maximum

or average) summarizing a group of output values (e.g. non-overlapping q × q regions) from the previous convolutional layer. See Figure 2.4 illustrating the convolution and pooling operations performed by a general CNN accepting an input image of size m × m (patch size) with b bands, employing r convolutional filters of size f × f followed by the pooling operation. Same as other artificial neural networks, the hidden layers of the CNN applies a non-linear activation as well.

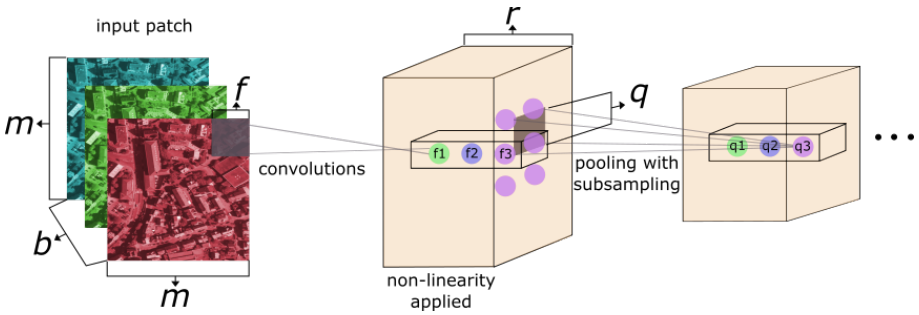


Figure 2.4 Illustration of the CNN input and convolution and pooling operations.

Convolutional neural networks were initially developed to recognize handwritten digits [81, 82]. But with recent advances in network design, optimization strategies, abundance of labeled data, and the advent of powerful graphical processing units (GPU) for computing, these networks continue to push forward state-of-the-art results in several computer vision tasks including: image classification, object detection, scene labeling/semantic segmentation, etc. Below we review several contemporary methods and applications contributing to the development of CNNs.

From the early networks like the LeNet-5 [82] with seven hidden layers, CNNs successfully trained recently have far greater depth than their early predecessors. Some popular examples are the AlexNet [76] with eight hidden layers, the VGGNet [130] with up to 19 hidden layers, the GoogLeNet [133] with 22 hidden layers, and the ResNet [61] with up to 1202 layers. Deeper networks have better representation ability—allowing the network to learn features of higher abstraction in the last convolutional layers, e.g. a composition of features in the previous layers—than their shallower counterparts. And these modern networks enumerated above [76, 130, 133, 61] empirically show that the depth of the network plays an important role in the latter's performance. But a deeper CNN will generally have more parameters to learn than a shallower

one: making deeper networks more prone to overfitting and more difficult to optimize.

All the networks mentioned above, except for GoogLeNet, uses the "usual convolution" with filter sizes fixed in each layer and smaller than the input. AlexNet uses relatively larger filters (11×11) in the first convolutional layers while using smaller filters (5×5 in the second and 3×3 in others) in the succeeding layers [76]. VGGnet uses small filters (3×3) all throughout its convolutional layers [130]. To keep the number of features relatively equal all throughout the layers, the number of filters of succeeding convolutional layer/s are generally a factor of n times the number of the preceding convolutional layer/s, where n is the downsampling factor of the pooling layer between the convolutional layers. GoogLeNet uses a special kind of convolutional layer they called "Inception module" [133]—a generalization of the operation applied by the Network in Network of Lin et al. [84]. Instead of applying the "usual convolution", the inception module applies convolutions of different filter sizes (5×5, 3×3, and 1×1) together with a maximum pooling operation within a single convolutional layer. Such that, at the end of the Inception module, the output of all sub-convolutions and maximum pooling are concatenated into a single output tensor. The Inception module also heavily applies 1×1 convolutions before the 5×5 and 3×3 convolutions as a means of dimension reduction. With the Inception architecture, GoogLeNet performs comparably well with networks of larger number of parameters (AlexNet has 15 times and VGGNet has 35 times more).

Another noteworthy convolutional network architecture are the residual networks [61]. These networks formulate convolutional blocks, composed of multiple convolutional layers, within a network as residual functions by propagating the input features of such blocks to their last layer through skip connections. More specifically, an original desired underlying mapping $\mathbf{y} = f_o(\mathbf{x})$ between the input $\mathbf{x}$ and output $\mathbf{y}$ can be residualized to the form of $\mathbf{y} = \mathbf{x} + f_r(\mathbf{x})$. Applying such an architecture allowed them to successfully train networks with considerable depths, i.e. having a number of hidden layers in the order of 10^2 to 10^3.

2.3 Recurrent Networks

Recurrent neural networks fall under the type of artificial neural network employing feedback/recurrent connection, i.e. connections forming a directed cycle. For example, the Jordan network [71] has connections from the output units back to the hidden units. Recurrent networks are particularly suited

for modeling sequential data. In [85], the authors reviewed the history and advances in recurrent architectures specifically for learning sequential data—e.g. image captioning, speech synthesis, etc. The authors in [115] applied a recurrent (Jordan variant) convolutional network for semantic segmentation of general images.

2.4 Deep Networks as Data-flow Graphs

We can generalize any variant of deep networks by seeing them as data-flow graphs—a graph representing how a set of input data are processed along a possibly branching chain of functions, in the end producing a final set of outputs. In this section, we describe elements of a convolutional network in terms of data-flow graphs. Using such a model, we define the networks by three elements: the sets of data they take as an input, the operations they perform in each function block, and the intermediate and final set of outputs they produce. The sequence of operations performed can be understood from the direction of the edges in the data-flow graph. Aside from these three key elements of data-flow graphs, details of a unique configuration and instance of a convolutional network are defined by its *hyperparameters* and *parameters* respectively. Hyperparameters are associated with the configuration of a network architecture and are set to fixed values before training the network. Parameters are values associated to a specific network instance and are learned during network training. Below we discus these three elements of data-flow graphs applicable to convolutional networks together with the parameters and hyperparameters associated with each element.

2.4.1 Input

A convolutional network receives as an input either the whole image itself to be classified or a subset of it, called an input patch.The dimension of this patch is defined by the patch size hyperparameter M and the number of bands B as shown in Figure 2.5. A network is generally trained over a group of image patches specified by the batch size N defining the number of image patches present in a single batch. A convolutional network accepts an $N \times B \times M \times M$ array of pixel values as an input (in the case of the image patches having equal height and width), N being the number of patches processed by the network in parallel. Aside from the input image patch, the corresponding reference image can also be considered as an input in terms of data-flow graphs since no operation precedes it.

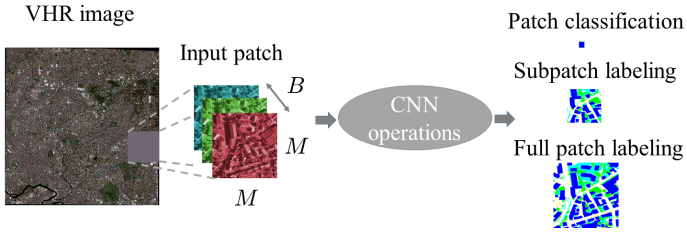


Figure 2.5 Illustration of CNN input and output types.

2.4.2 Operations

Convolutions are the main operations used by convolutional networks. A convolution applies a linear operation on an input image/feature map using a set of K' learnable kernels. The kernel values makes up the main chunk of network parameters. Applying a kernel $\mathbf{w}$, composed of a $K \times K' \times G \times G$ array of learnable parameters, on a $K \times H \times W$ input feature map $\mathbf{x}$, where G is the kernel size, K is the number of kernels in each set of kernels, and H and W are the height and width of the feature map, produces a $K' \times H' \times W'$ output feature map $\mathbf{x}'$. The output at the i' row and j' column of the k' feature map is given by:

$$\mathbf{x}'_{k'i'j'} = \sum_{k=1}^{K} \sum_{p=1}^{G} \sum_{q=1}^{G} \mathbf{x}_{kij} \cdot \mathbf{w}_{kk'pq} + \mathrm{b}_{k'} \tag{2.4a}$$

$$i = i' + p - \lceil \frac{G}{2} \rceil \tag{2.4b}$$

$$j = j' + q - \lceil \frac{G}{2} \rceil \tag{2.4c}$$

where $\mathrm{b}_{k'}$ is the learnable bias parameter associated with the k' feature map. The width and height of the output feature map are given by:

$$H' = \lfloor \frac{H - G + 2Z}{S} + 1 \rfloor \tag{2.5a}$$

$$W' = \lfloor \frac{W - G + 2Z}{S} + 1 \rfloor \tag{2.5b}$$

where the *zero-padding* Z is the number of rows and columns of zeros added to the border of the input feature map and the convolutional *stride* S is the number of units separating contiguous receptive fields of the kernel on the input feature

map. Equation 2.5 implies that both Z and S have the same values for the two spatial dimensions (row-wise and column-wise) of the feature maps. The case of uneven zero-padding and strides can easily be derived from the same equation. If $G = H$ then a convolution equivalently becomes the same operation applied by fully-connected feedforward networks—where each unit in the succeeding layer has an independent weight connected to every unit of the previous layer. A standard convolution, however, would have local-connectivity where $G < H$ effectively applying exactly the same set of $\mathbf{w}$ in different locations of the input feature map. In this setup, an implementation of convolution can be viewed as a moving window operation resulting to elementwise multiplication of the kernel values and the values of the units within its receptive fields. This effect of reusing the same set of weights in different parts of the input is called parameter sharing, which reduces the number of parameters when compared to a fully-connected variant. Parameter sharing also reflects the prior knowledge that we expect similar patterns to be present in different areas of an input feature map, e.g. a vertical edge might be present both in the upper left corner as well as the lower right corner of an image. To preserve the spatial dimensions of the input ($H' = H$ and $W' = W$), a conventional approach would be to set $S = 1$ and $Z = \frac{G-1}{2}$ if G is odd and $Z = \frac{G}{2}$ if G is even. The authors in [50, pp. 342–352] discusses variants of the basic convolution such as unshared convolution (also called locally-connected layer) and tiled convolution. In an unshared convolution, the connection is local ($G < H$) but the kernels are never reused—hence, having different sets of $\mathbf{w}$ for each location in the input. Tiled convolution compromises between shared (basic) and unshared convolution such that: instead of totally sharing the same set of $\mathbf{w}$ for all parts of the images, it applies T separate sets of $\mathbf{w}$ every $(S{\times}T)_{th}$ unit. Tiled convolution becomes shared convolution when $T = 1$; and becomes unshared convolution when $T = H'{\times}W'$. Another noteworthy variant is dilated convolution used by authors in [23] to arbitrarily increase the size of the kernel—from $G{\times}G$ to $(G{\times}D){\times}(G{\times}D)$ by filing in zeros in-between—without further increasing computational burden.

Nonlinearity is applied after the linear operation of a convolution. Since applying a series of linear operations can be reduced to a single linear operation, an elementwise nonlinear function applied between each convolution allows the network to learn more complex input to output mapping. A common choice is the rectifier function

$$\mathbf{x}'_{i'j'k'} = \max(0, \mathbf{x}_{ijk}) \tag{2.6}$$

or a variation of it [93, 59, 29]. The first convolutional network LeNet-5 [82] employs a scaled hyperbolic tangent (tanh) function as its nonlinear

operation. Another previously commonly used activation is the logistic sigmoid function. However, since both nonlinearities saturates at their tails—the former having nearly-zero gradients at the latter, training deep networks may cause slow convergence or even non-convergence of the learning process due the so called "exploding/vanishing gradient" problem [152]. Modern networks [76, 130, 133, 61], on the other hand, heavily use the rectifier function which does not saturate on the positive domain. Several works [48, 93, 160] observed that networks using rectified linear units (ReLU) perform better than those using their saturated counterparts (tanh or sigmoid). The potential drawback of using ReLU is having zero gradient for every inactive unit—hence, permanently "killing" a unit (setting it to zero). To address this drawback, a number of functions generalizing ReLU were proposed: leaky ReLU [93], parametric ReLU [59], randomized ReLU [152], and exponential linear unit [29]. Leaky ReLU allows a fixed small gradient δ to pass through units with negative activation values [93]. Instead of using a fixed value, parametric ReLU learns δ as an additional parameter for each feature map in the network [59]. Randomized ReLU samples δ from a uniform distribution during training, and uses a fixed value in testing [152]. Exponential linear units assign an exponentially saturating value to the negative part of the rectifier—effectively improving training convergence and generalization of the networks [29]. Maxout units [51] further generalizes the ReLU by taking the maximum across the channels of the input feature map at the same spatial location. Maxout units can learn piecewise linear convex activation function [51]. A variant of maxout, probout units [132] samples the same spatial location across channels of the input feature map based on a multinomial distribution given by the normalized activation values in the same location.

Pooling takes an aggregate of values over local regions of the input. A common choice of a pooling function is the average or maximum function. We can view pooling the same way as we view convolutions—where a moving $G \times G$ window of stride S is applied to the input feature map. However, in contrast to convolution, a basic pooling does not have any learnable parameters. Originally, pooling was used to give the network a small degree of translation invariance by summarizing values of the input on non-overlapping windows ($S = G$)—also downsampling the input by a factor of S, with proper zero-padding. Downsampling increases the receptive fields of succeeding convolutions while decreasing the computational burden by reducing the spatial dimensions of the output feature maps. Depending on the dataset, networks using maximum pooling may outperform those using average pooling; and for some other dataset an optimal pooling could be somewhere in between the two mentioned operations [17]. Realizing this, [54] proposes a parametrized pooling by taking

the l_p norm of units over a pooling region. l_p pooling reduces to average pooling when the order of the norm $p = 0$ and max pooling when $p = \infty$. In [54], p is a parameter of the model and is learned for each pooled unit in the network. Inspired by the regularization method *Dropout* [63], [155] and [158] proposes two pooling operations: mixed pooling and stochastic pooling respectively. Mixed pooling assigns a random binary variable λ to each pooling region—performing maximum pooling for $\lambda = 1$ and average pooling for $\lambda = 0$ [155]. Stochastic pooling, on the other hand, randomly samples each pooling region based on the multinomial distribution given by the normalized activation values within the pooling region [158]. Both these Dropout-inspired pooling also promotes regularization in CNNs. He et al. introduces the pooling strategy called spatial pyramid pooling [60] to address the "artificial problem" most CNN implementations have: requiring fixed input sizes. Spatial pyramid pooling aggregates features from convolutional layers into local spatial bins with sizes proportional to the input feature map—hence, the number of bins is fixed regardless of the size of the input feature map [60].

Upsampling operations are applied to increase the spatial dimensions of input feature maps. Upsampling is important specifically if the network needs to produce output predictions of the same size as the input, i.e. we want to produce a label for each pixel in the $M \times M$ input patch. One way to upsample is by employing resampling techniques such as nearest neighbor or bilinear interpolation [23]. The original fully convolutional network (FCN) [87] learns the upsampling operation using *backwards convolution* (or more technically fitting called *transposed convolution*). Backwards convolution is equivalent to the operation performed when calculating the gradients of a convolutional operation. Convolutions can be efficiently expressed as a matrix operation and its gradients can be computed by multiplying the backpropagated error of the succeeding layer by the transpose of the matrix representing the convolution—hence, the name transposed convolution. Another approach called *unpooling* [159, 6] can be used to upsample input feature maps by saving the row and column indeces of max-pooling operation with downsampling and copying the values of the input to corresponding indices of a higher resolution output.

Identity operations are used to propagate information from lower-level feature maps to higher-level ones and are called *skip connections* in [87]. It can also be used to apply a recurrent connection within a network as done by [115].

Merging combines two or more sets of feature maps in a network either by addition or by concatenation. Addition is an elementwise operation performed between feature maps—adding each unit with corresponding indices—hence, all the three dimensions (K, H, W) must be the same for all inputs [87]. Concatenation stacks the input feature maps depth-wise—hence, only the

spatial dimensions (H, W) must be the same. The authors in [87] used addition to combine lower-level features with higher-level ones.

Other technique-specific operations such as masking used in Dropout and parametrized normalization used in *Batch Normalization* (BN) [68] may also be used between convolutions to improve the training and generalization capability of the networks. Index-searching funtions, such as arg max that returns the index of the maximum values along a dimension, and comparison operations are used in the evaluation of objective loss and classification accuracy. More details on these techniques can be found in Section 2.5.

2.4.3 Outputs

In data-flow graph terms, the outputs of a convolutional network consist of all the intermediate feature maps, the final class score maps, and the corresponding loss and accuracy calculated using the class score maps and the reference labels. Characteristics of resulting intermediate feature maps were discussed in Section 2.4.2 while calculation of objective loss and classification accuracy will be discussed in the succeeding sections. Final class score maps correspond to the units in the last layer of a neural network and its dimension depends on how the task is defined. Authors in [146] categorize the approaches to this task into three variants: 1) patch classification, 2) subpatch labeling, and 3) full patch labeling. In patch classification, we assign a single label to the patch, i.e. the label corresponds to the class of the central pixel of the patch [12, 146, 98] (see Figure 2.5). In subpatch labeling, we assign labels on a smaller part of the patch corresponding to the area near the center of the patch [146]. Finally, in full patch labeling, we assign labels to all the pixels in the patch [87, 128, 6, 146, 113]. The last method, aside from being more efficient, also decouples the limit of the input patch size to the number of downsampling operations in the network. The class score map dimensions are $C\times1\times1$, $C\times M_{sub}\times M_{sub}$, and $C\times M\times M$ where $1 < M_{sub} < M$ for the patch classification, subpatch labeling, and full patch labeling respectively.

2.4.4 Parameters and hyperparameters

Table 2.1 summarizes the parameters and hyperparameters in the different elements of a data-flow graph representing a convolutional network. Only convolutional layers (including transposed convolution) are parametrized. Other values and functions—such as the activation f_a, pooling f_p, upsampling f_u, and merging f_m function—are hyperparameters and are fixed beforehand.

Table 2.1 Parameters and hyperparameters in a CNN

Graph element	Parameters	Hyperparameters
Input	none	M
Convolution[a]	$\mathbf{w}$, $\mathbf{b}$	G, S, Z
Nonlinearlity[a]	none	f_a
Pooling	none	G_p, S_p, Z_p, f_p
Upsampling	$\mathbf{w}$, $\mathbf{b}$ [b]	f_u
Merging	none	f_m

[a] Basic convolution and non-parametrized activation functions

[b] When transposed convolution is used

$M$is the input patch size

$\mathbf{w}$, $\mathbf{b}$ are the kernel and bias weights

G, S, Z are the kernel size, stride, and zeropadding

f_* symbolizes a function and subscripts $_{a,p,u,m}$ corresponds to activation, pooling, upsampling, and merging

2.5 Training Deep Networks

We determine the values of the parameters of the network in a step called *training phase*. In a supervised learning setting, where we have available reference labels corresponding to our training samples, we search for the "best" possible values of our network parameters by showing the network sets of examples where the network compares its predictions (based on the examples it has seen) against targets (reference labels) associated with the examples. We formalize the comparison by defining an objective function. We train the network by minimizing an objective function in terms of the parameters of the network. For classification involving C classes, a cross-entropy loss function is often used given by:

$$E_N(\mathbf{w}) = -\sum_{n=1}^{N} \mathbf{t}_n \cdot \log(\mathbf{y}_n) \tag{2.7}$$

where E is the loss function value evaluated over N samples, $\mathbf{t}_n$ is a binary vector encoding the the target class labels (with the index corrresponding to a class having a value of 1 and 0 otherwise), $\cdot$ denotes the dot product, and $\mathbf{y}_n$ is the class score maps of a sample n calculated using a *softmax* activation

function:

$$\mathbf{y}_{kij} = \frac{\exp(\mathbf{x}_{kij})}{\sum_{c=1}^{C} \exp(\mathbf{x}_{cij})}. \tag{2.8}$$

In this equation, $\mathbf{y}$ is the softmax score and $\mathbf{x}$ is the last set of feature maps containing unnormalized class scores at location ij.

We train the network by minimizing a specified objective function. The most common method to minimize the objective function is an iterative gradient-based optimization technique called backpropagation with gradient descent [120] or a variant of it. Backpropagation computes the derivative of the loss function with respect to the learnable network parameters and gradient descent updates the weights by adding a value proportional to the negative of the gradients. The weight update $\Delta\mathbf{w}$ is obtained by:

$$\Delta\mathbf{w}(\tau) = -\eta(\tau)\frac{\partial E(\tau)}{\partial \mathbf{w}(\tau)} + \alpha\Delta\mathbf{w}(\tau - 1) \tag{2.9}$$

where $\frac{\partial E}{\partial \mathbf{w}}$ is the vector of gradients, η is the learning rate, and α is the momentum hyperparameter at epoch τ. An epoch is defined to be the number of iterations required for the network to compute gradients using all training samples, while a single iteration is a one-time evaluation and application of Equation 2.9. The evaluation of Equation 2.9 can be decomposed into two different steps: the *forward pass* and *backward pass* of the network. A forward pass consist of applying all the series of operations a CNN has to calculate the objective function value E. A backward pass, on the other hand, computes the gradients and correspondingly produce an evaluation of Equation 2.9. The learning rate and momentum are hyperparameters of the optimizer. The learning rate hyperparameter defines the proportion of the gradient values that we need subtract from the previous gradients—analogous to the "size of the step" we take in the parameter space to search the latter's optimal values. The momentum method [116] accelerates the convergence of the optimizer by forcing it to the same direction as the previous gradient update—effectively "dampening oscillations" in regions of the parameter space with problematic curvatures and gradients.

The basic variant of backpropagation with gradient descent evaluates the weight update over the whole set of training samples and is called *batch* gradient descent. However, in practice, the batch version of gradient descent is often too computationally expensive since we need to evaluate the gradients over all training samples before calculating the final weight updates. Hence, we

often use a *stochastic* version of the gradient descent where we approximate the weight updates over randomly sampled subsets (called *mini-batches*) of the training set. We can infer predictions from the final trained network instance by truncating the loss evaluation in the computational graph and taking the index of the maximum class score map value along the class score dimension by

$$\mathrm{y}_{ij} = \arg\max_{c} \mathbf{y}_{cij} \tag{2.10}$$

where $\mathbf{y}$ and y are the class score and prediction for location ij respectively.

Advances in optimization methods address the problem of underfitting. Underfitting happens when a learning algorithm gets stuck in a poor set of parameter values—hence, performing (almost) equivalently badly in both training and testing phase. Several proposed solutions to overcome the underfitting problem, aside from stochastic gradient descent [16], are: proper weight initialization [47], batch normalization [68], and shortcut connections [61]. In [47], the authors proposed to initialize the weights with values randomly sampled from a Gaussian distribution with variance $2/(\mathrm{n}_{in} + \mathrm{n}_{out})$, where n_{in} and n_{out} are the number neurons in the preceding and succeeding layers. Batch normalization [68], as the name implies, transforms the activations (by batch) of a preceding feature map to follow a normal distribution with $\mathcal{N}(0, 1)$—instead of just performing normalization of the whole training set. In [61], the authors employ shortcut connections in the form of identity mapping within hidden layers of deep networks.

2.6 Regularizating Deep Networks

Deep networks are often prone to overfit the training set. Overfitting occurs when a model reports high accuracy during training but performs poorly on unseen test data. Regularization approaches address the overfitting problem using three common methods: *data augmentation*, *weight decay*, and *early-stopping*. Data augmentation technique increases the number of training samples by permuting them with applicable rotational and/or translational transformations. Data augmentation helps the network to learn relevant invariances that may be present in the input. Weight decay modifies the loss function by

$$Q(\mathbf{w}) = E(\mathbf{w}) + \lambda \|\mathbf{w}\|_2^2 \tag{2.11}$$

adding a penalty proportional to the square of the l_2-norm of the weight vector $\mathbf{w}$. The weight decay λ hyperparameter controls the contribution of this penalty

to the loss function. Such penalization promotes weight values near the origin of the parameter space—hence, allowing more uniform/smoother values. Early stopping prematurely stops the training when a criterion measured from a validation set is met. For example, we can stop the training when the value of the loss function or the classification accuracy evaluated on the validation set did not change by more than 1% for the past 5 epochs.

Two more recent algorithms addressing the overfitting problem are: dropout [63] and dropconnect [148]. Dropout basically randomly drops a unit within a hidden layer by a probability $1 - \psi$, where ψ defines the chance of retaining a unit during training. In testing phase, the weights of the network are multiplied by a factor of ψ. It is equivalent to sampling a binary vector $\mathbf{d}$ whose elements are sampled from a Bernoulli distribution parametrized by ψ. Dropout effectively samples different architectures of the network at training time indirectly creating an ensemble of network. Dropout has been empirically shown to improve the ability of deep networks to generalize on unseen data set. In dropconnect, instead of zeroing out the hidden units, the connections between units are randomly dropped instead. Authors in [141] observed that the standard dropout method does not help in regularization when applied to convolutional layers and, hence proposed an new method called SpatialDropout.

We summarize common learning and regularization hyperparameters and parameters in table 2.2.

Table 2.2 Learning and regularization parameters and hyperparameters

Method	Parameters	Hyperparameters
Stochastic Gradient Descent	none	η, α, N, $\mathcal{T}$
Batch Normalization	γ, β	none
Weight decay	none	λ
Early-stopping	none	stopping criteria
Dropout	none	ψ

η, α, N, $\mathcal{T}$ are the learning rate, momentum, batchsize, and number of epochs
γ and β are the scaling and shift parameters [68]
λ is the weight decay rate
ψ is the dropout rate.

FuseNet: End-to-end Multispectral VHR Image Fusion and Classification

3

Abstract

Classification of very high resolution (VHR) satellite images faces two major challenges: 1) inherent low intra-class and high inter-class spectral similarities and 2) mismatching resolution of available bands. Conventional methods have addressed these challenges by adopting separate stages of image fusion and spatial feature extraction steps. These steps, however, are not jointly optimizing the classification task at hand. We propose a single-stage framework embedding these processing stages in a multiresolution convolutional network. The network, called *FuseNet*, aims to match the resolution of the panchromatic and multispectral bands in a VHR image using convolutional layers with corresponding downsampling and upsampling operations. We compared *FuseNet* against the use of separate processing steps for image fusion, such as pansharpening and resampling through interpolation. We also analyzed the sensitivity of the classification performance of *FuseNet* to a selected number of its hyperparameters. Results show that *FuseNet* surpasses conventional methods.

This chapter is based on:

J. R. Bergado, C. Persello, and A. Stein. FuseNet: End-to-end multispectral VHR image fusion and classification. 2018 IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2018 - Proceedings, pp. 2091-2094, Jul 2018.

J. R. Bergado, C. Persello, and A. Stein. Recurrent multiresolution convolutional networks for vhr image classification. IEEE Transactions on Geoscience and Remote Sensing, 56(11):6361–6374, Nov 2018.

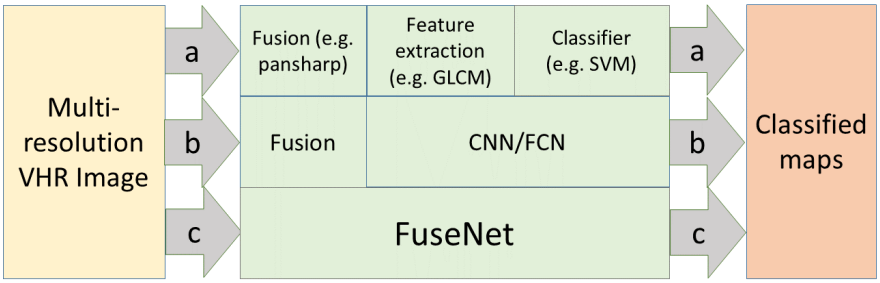


Figure 3.1 Different pipelines for classifying multiresolution VHR images.

3.1 Introduction

Classification of very high resolution (VHR) satellite images presents two major challenges: 1) inherent low intra-class and high inter-class spectral similarities and 2) mismatching resolution of available bands. The first challenge is often addressed by extracting spatial-contextual features from the image such as texture-describing measures, e.g. gray level co-occurrence matrix (GLCM) and local binary patterns (LBP) [109] or products of morphological operators [40] that are expected to reduce spectral class ambiguities. The second challenge is dealt with pansharpening and interpolation-based resampling techniques used to fuse images of different resolutions. A typical approach to classification of a multiresolution VHR satellite image would then be as shown in Figure 3.1 (a). These additional steps to address problems in classifying a multiresolution VHR satellite image are disjoint from the supervised classifier, and hence, not optimized for the task at hand.

Deep learning offers a framework to build *end-to-end* classifiers by directly learning the predictions from the inputs with minimal or no separate pre-classification steps. Convolutional neural networks (CNN), for instance, integrate the feature extraction step within the training of the supervised classifier and have performed better than intermediate handcrafted features [12, 98]. Recently, a patch-based CNN [98] and a fully convolutional network (FCN) [113], utilizing pansharpening for image fusion, were used to detect informal settlements from a multiresolution VHR satellite image. Both works have addressed the classification challenges as in Figure 3.1 (b). In this paper, we present a novel single-stage network performing image fusion and classification of a multiresolution VHR satellite image in an end-to-end fashion as in Figure 3.1 (c).

3.2 Data and Methods

We propose a multiresolution convolutional network, called FuseNet, to perform an end-to-end image fusion and classification of a multi-resolution VHR satellite image. FuseNet is built on top of a fully convolutional network architecture learning to: 1) fuse panchromatic (PAN) and multispectral (MS) bands of a VHR satellite image, 2) extract spatial feature, and 3) classify land cover classes.

FuseNet is specifically designed for VHR satellite images with PAN band and MS bands having a ground sampling distance ratio of four (e.g. Quickbird, Worldview 2/3, Pleiades, Ikonos). This architecture can be generalized to fuse any number of images with different spatial resolutions and any number of bands. It accepts two sets of input: an image patch of dimensions $N \times 1 \times 4M \times 4M$ taken from a PAN image and another patch of dimensions $N \times 4 \times M \times M$ taken from corresponding locations in the MS image. It performs two series of convolution, nonlinearity, and maximum pooling with downsampling to the PAN image patches such that the spatial dimensions of the intermediate feature maps match the spatial dimensions of the MS image patches. The nonlinear operations use an exponential linear activation function [29]. The second input is linearly projected in k dimensions using 1×1 convolutions such that k matches the number of intermediate feature maps extracted from the first set of input. This ensures that succeeding feature maps extract the same number of pattern variations from both sets of inputs. FuseNet merges the linear projection of the MS image patches with intermediate feature maps extracted from the PAN image patches via a concatenation operation.

Additional series of convolution, nonlinearity, and maximum pooling with downsampling operations are applied to the merged feature maps thus producing a set of feature maps with the smallest spatial dimensions—called a *bottleneck*. FuseNet then upsamples the bottleneck back to the resolution of the PAN input image patches using transposed convolutions. The resulting set of feature maps is linearly projected again using 1×1 convolutions such that the number of feature maps matches the number of classes C. FuseNet applies a softmax activation to calculate normalized class score maps and couples those with a cross-entropy loss function (see Equation 5.2).

Figure 3.2 The general architecture of FuseNet with skip connections (FuseNet$_{skip}$). FuseNet accepts two streams of input: one from PAN image patches and another from MS image patches. It applies convolutional and pooling layers with downsampling to extract spatial features and at the same time match the resolution of the two streams of input. Similar operations are performed to the output of the merged streams of input arriving at a feature map with smallest spatial dimensions (bottleneck). From there, upsampling operations using transposed convolutions are performed to restore the resolution back to the resolution of the PAN image patches. Skip connections are implemented using appropriate upsampling and linear projections.

Table 3.1 Detailed operations of FuseNet$_{low}$.

FuseNet$_{low}$	
$\mathbf{x}_{PAN}(1\times4M\times4M)$	$\mathbf{x}_{MS}(4\times M\times M)$
conv13-16 maxpool conv7-32 maxpool	conv1-32
IFM1 ($32\times M\times M$)	IFM2 ($32\times M\times M$)
concat	
IFM3 ($64\times M\times M$)	
conv3-64 maxpool conv3-128 maxpool	
BFM ($128\times M/4\times M/4$)	
ups2-128 ups2-64 ups2-32 ups2-16 conv1-6	
IFM4 ($6\times4M\times4M$)	
softmax	

Table format as in [130].
$\mathbf{x}_{PAN}$ and $\mathbf{x}_{MS}$ denote input patches from the PAN and MS images, respectively.
IFM and BFM corresponds to intermediate and bottleneck feature maps respectively.

This configuration of FuseNet is called FuseNet$_{low}$ because it performs fusion at the lower (MS image) resolution. We also tested a network, called FuseNet$_{skip}$, adding skip connections to lower-level feature maps of FuseNet$_{low}$ [87]. Additionally, we experimented with a version of FuseNet performing fusion at the resolution of the PAN image, called FuseNet$_{high}$ which is more similar to pansharpening as it upsamples the MS image patches first before fusing them with the PAN image patches. Table 3.1 shows details of the operations, including dimensions of intermediate output feature maps used by FuseNet$_{low}$. Figure 3.2 shows the the skip connections in the FuseNet$_{skip}$ variant.

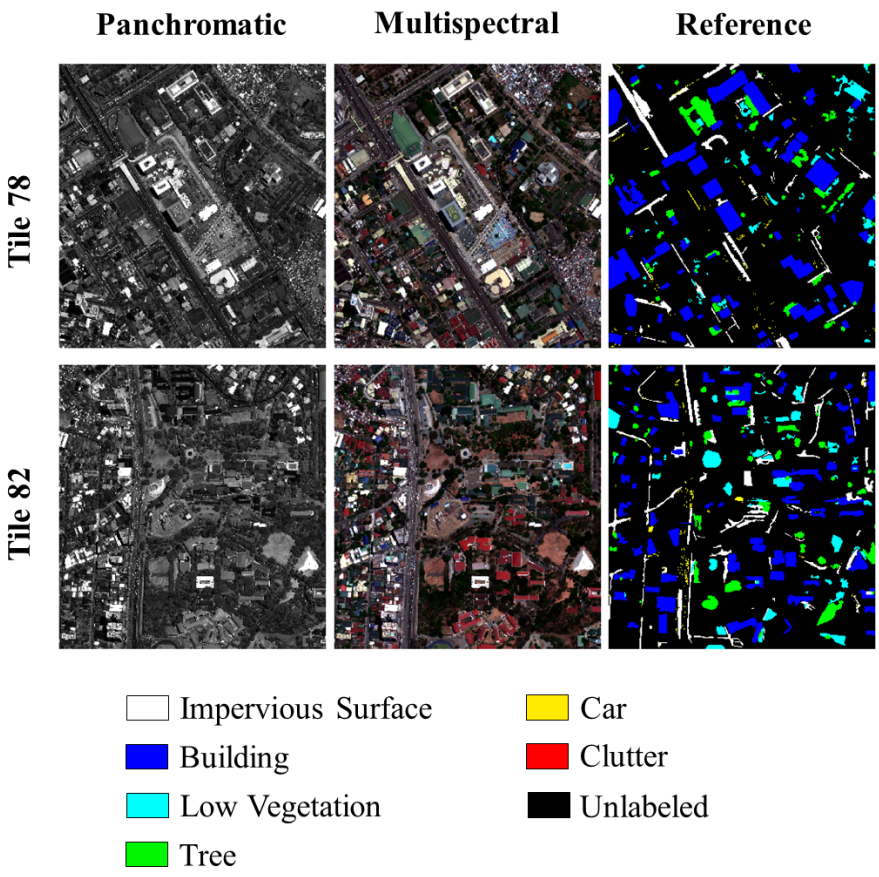


Figure 3.3 PAN, MS, and reference images in the tiles used for testing.

We evaluated the proposed network for a land cover classification of a dataset covering Quezon City, Philippines. The dataset is composed of a Worldview-03 satellite image acquired on 17^{th} April 2016 and manually prepared reference images for five chosen tiles (subsets) of the satellite image. The satellite image has a PAN band of 0.3 m resolution and four MS bands (near-infrared, red, green, and blue) of 1.2 m resolution.

The satellite image was first divided into regularly-sized image tiles. PAN image tiles have a dimension of 3200×3200 pixels, while MS image tiles have a

dimension of 800×800 pixels. Five non-adjacent tiles were sparsely labeled—annotating a pixel with a label belonging to one of the following six classes: 1) impervious surface, 2) building, 3) low vegetation, 4) tree, 5) car, and 6) clutter. Two of the five labeled tiles were used for training (tiles 100 and 105), one for validation (tile 45), and the remaining two for testing (tiles 78 and 82). Figure 3.3 shows the two tiles used for testing.

We compared FuseNet with baseline methods using pansherpening and using bilinear interpolation to match the resolution of the MS image patches to the resolution of the PAN image patches, called $\text{Net}_{pansharp}$, SegNet [6], $\text{Net}_{pan-cnn}$, and $\text{Net}_{bilinear}$, respectively. $\text{Net}_{pansharp}$ applies the Gram-Schmidt pansharpening technique, SegNet uses the first three principal components of the inputs of $\text{Net}_{pansharp}$, while $\text{Net}_{pan-cnn}$ adapts the CNN-based pansharpening method proposed in [126]. Only the pansharpened image is fed as an input into $\text{Net}_{pansharp}$, SegNet, and $\text{Net}_{pans-cnn}$. In contrast, $\text{Net}_{bilinear}$ upsamples the resolution of the MS image to match the resolution of the PAN image using bilinear interpolation. The upsampled MS images are then merged with the PAN image using concatenation. The architecture of the network after the fusion, except for SegNet, is kept the same to have a fair comparison among the different methods.

3.3 Results and Discussion

Table 3.2 Comparison of fusion approaches

Network	OA (%)	κ (%)	AA (%)	F1 (%)
$\text{Net}_{bilinear}$	84.76	78.70	81.99	77.48
$\text{Net}_{pansharp}$	86.87	81.53	82.76	77.86
$\text{Net}_{pan-cnn}$ [126]	87.88	82.69	84.58	72.45
SegNet [6]	88.11	83.17	83.96	77.01
FuseNet_{high}	88.03	83.18	89.79	79.06
FuseNet_{low}	91.63	88.03	92.91	**82.90**
FuseNet_{skip}	**91.90**	**88.43**	**93.46**	81.74

Table 3.2 shows the results of accuracies comparing different fusion approaches. FuseNet_{skip} scores the highest in all the four numerical metrics, except for F1 where FuseNet_{low} scores the highest. Correspondingly, FuseNet_{low}, the architecture from which FuseNet_{skip} was derived, outperforms all the other networks, except for FuseNet_{skip} itself. Observing each metric: FuseNet_{low}

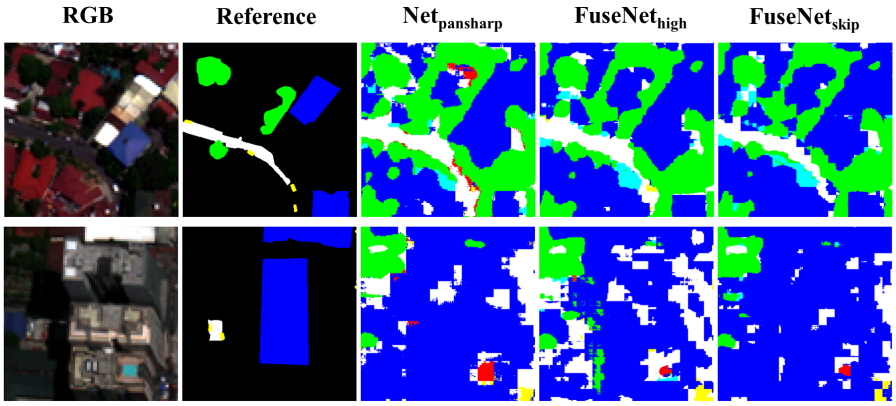


Figure 3.4 Results of selected FuseNet variants and baseline methods.

gains about 3–6% in OA, 4–9% in κ, 3–10% in AA, and 3–10% in F1 against the other baselines. FuseNet$_{skip}$ further increases the numerical results of FuseNet$_{low}$ for the first three metrics by about 0.2–0.5%, but degrades the F1 by about 1.2%.

We notice that: 1) learning fusion can improve the classification of PAN and MS VHR images with different resolutions; 2) fusing at the scale of the image with lower resolution results in better classification than performing fusion at the scale of the image with higher resolution. The first point demonstrates our expected effectiveness of coupling and learning the fusion operation within a supervised classifier. Regarding the second point, introducing upsampling layers early in the network (FuseNet$_{high}$) may produce artifacts that can degrade its performance.

Figure 3.4 shows the classification maps from selected FuseNet variants and baseline methods. The most noticeable misclassifications are found in large and high-rise buildings and overpassing roads. The facades and rooftops of the buildings are often mistaken to be impervious surfaces by the classifiers, while overpassing roads are mistaken to be a building. These regions can appear to have similar spectral characteristics and can only be distinguished by presence of other indications such as appearing to be elevated. Manually distinguishing arguably vaguely-defined classes such as low-vegetation and impervious surface can also be problematic, especially in the PAN image, with the lack of ancillary information such as elevation. The cars are also generally misclassified by all the classifiers. This is, aside from being underrepresented in terms of the number

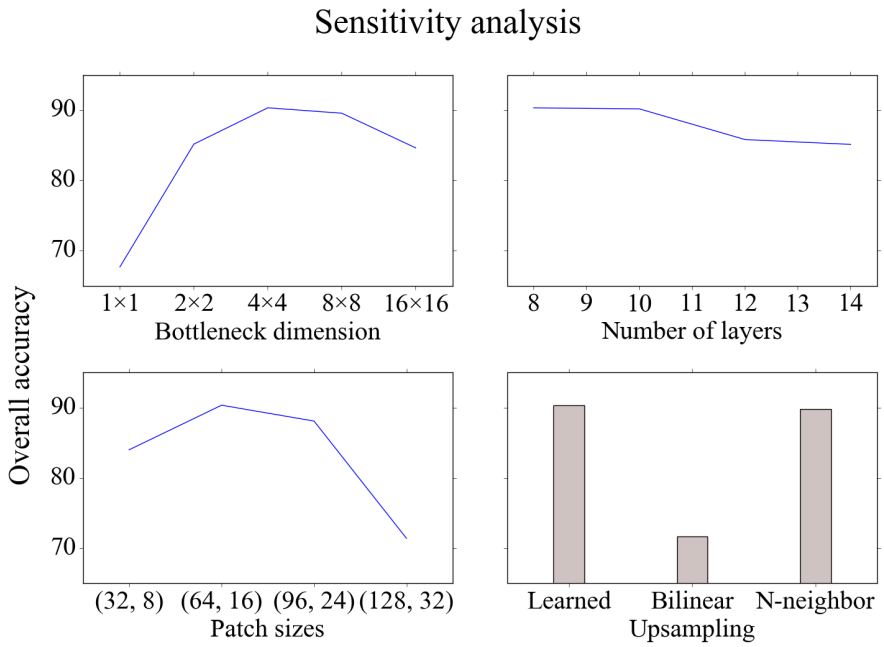


Figure 3.5 Plots showing the results of FuseNet sensitivity analysis.

of labeled pixels, due to the lack of spatial resolution of the MS bands and the spectral similarity of cars with other classes such as impervious surface and buildings in the PAN band. Overall, FuseNet$_{skip}$ has less errors in the facade of large buildings and provides a better delineation of classes with irregular boundaries such as trees and low-vegetation—providing the best classification results.

3.3.1 Sensitivity Analysis

Fig. 3.5 shows the results of a sensitivity analysis performed on four chosen hyperparameters of FuseNet: 1) bottleneck feature map dimensions, 2) number of convolutional layers in the downsampling part of the network, 3) input patch sizes, and 4) upsampling methods. We got the highest validation accuracy of 90.35% using a bottleneck feature map dimension of 4×4 pixels. Decreasing the dimension below its optimum severely degrades the classification resulting to large uniform areas producing stamp-like patterns especially at the 1×1

level. Increasing the dimensions produces much noisier classification. Fixing the bottleneck size dimension to 4×4 and further increasing the number of convolutional layers without downsampling did not produce any improvements in the validation accuracy. Hence, the results show that with only eight convolutional layers with downsampling, we can learn enough contextual information for the most accurate classification.

We found the optimal patch sizes to be equal to 64×64 for the PAN image patches and 16×16 for the MS image patches. Further increasing the patch sizes results in overclassification of a single class impervious surface. Increasing the patch size also increases the proportion of frequently occurring classes in the training sample, possibly resulting into overclassification. Lastly, we noted that the use of transposed convolution for learned upsampling performs better than the use of interpolation for fixed upsampling. This result supports the expected flexibility of empirically learning upsampling directly from the data.

3.4 Conclusion

In this paper, we presented a multiresolution convolutional network named FuseNet to classfiy a VHR satellite image. The operations for fusing the bands with different resolutions are learned within convolutional layers with corresponding downsampling and upsampling operations to match the resolution of the images. Results show the advantages of incorporating image resolution matching within the training of the classifier. To this end, we provided a single-stage classification pipeline incorporating image fusion and feature extraction combined in a convolutional network trained in an end-to-end manner.

ReuseNet: Integrating Contextual Label Information via Network Recurrence

4

Abstract

Classification of very high resolution (VHR) satellite images has three major challenges: 1) inherent low intra-class and high inter-class spectral similarities, 2) mismatching resolution of available bands, and 3) the need to regularize noisy classification maps. Conventional methods have addressed these challenges by adopting separate stages of image fusion, feature extraction, and post-classification map regularization. These processing stages, however, are not jointly optimizing the classification task at hand. In this study, we propose a single-stage framework embedding the processing stages in a recurrent multiresolution convolutional network trained in an *end-to-end* manner. The feedforward version of the network, called *FuseNet*, aims to match the resolution of the panchromatic and multispectral bands in a VHR image using convolutional layers with corresponding downsampling and upsampling operations. Contextual label information is incorporated into *FuseNet* by means of a recurrent version called *ReuseNet*. We compared *ReuseNet* against the use of separate processing steps for both image fusion, e.g. pansharpening and resampling through interpolation, and map regularization such as *conditional random fields*. We carried out our experiments on a land cover classification task using a Worldview-03 image of Quezon City,

This chapter is based on:
J. R. Bergado, C. Persello, and A. Stein. Recurrent multiresolution convolutional networks for vhr image classification. IEEE Transactions on Geoscience and Remote Sensing, 56(11):6361–6374, Nov 2018.

Philippines and the ISPRS 2D semantic labeling benchmark dataset of Vaihingen, Germany. *ReuseNet* surpasses the baseline approaches in both quantitative and qualitative results.

4.1 Introduction

Classification of very high resolution (VHR) remotely sensed images allows us to automatically produce maps at a level of detail comparable to conventional in-situ mapping methods. Due to the high spatial resolution of such images, automated classification comes with a set of challenges. One challenge is the inherent low intra-class and high inter-class spectral similarities, inhibiting discrimination of the classes of interest. Conventional methods address this challenge by extracting spatial-contextual features from the image such as texture-describing measures, e.g. gray level co-occurrence matrix (GLCM) [57] and local binary patterns (LBP) [109], or products of morphological operators [106, 40]. This step is crucial for obtaining discriminative features and accurate classification. However, such feature extraction methods are often disjoint from the supervised classifier, and, hence, not optimized for the task at hand. Deep learning offers a framework to build end-to-end classifiers—directly learning the predictions from the inputs with minimal or no pre-classification and post-classification steps. Features automatically extracted by deep learning based classifiers such as convolutional neural networks (CNN) [82] perform better than intermediate handcrafted features [12, 98]. These networks automatically learn spatial-contextual features directly from the input VHR image—effectively integrating the feature extraction step into the training of the classifier as shown in Figure 4.1 (b). The design of network architecture, inspired by the model of the visual cortex [66], makes CNN suitable for image analysis and land cover classification.

Literature shows that classification accuracy can be improved by using post-classification spatial regularization [23, 110, 150]. Methods employing graphical models, such as conditional random fields (CRF) and Markov random fields (MRF), provide a way to perform this spatial regularization step. Similar to the two pre-classification steps described above, a post-classification map regularization technique adds another step independent of the training of the classifier itself—further including a separate objective function to be optimized. For classifying a multiresolution VHR image, a typical classification pipeline would be composed of three main stages: a pre-classification step performing image fusion and feature extraction, a supervised learning algorithm performing the classification, and a post-classification step regularizing the maps obtained

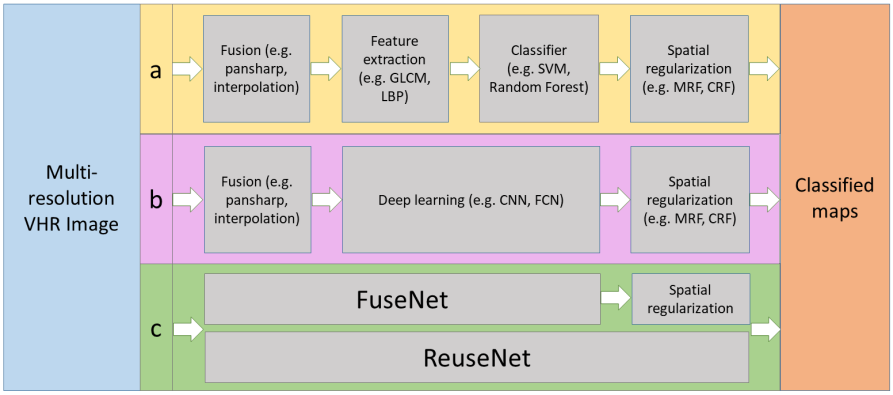


Figure 4.1 Illustration comparing a standard (a), state-of-the-art (b), and proposed (c) piplelines for classifying multiresolution VHR images.

from the supervised classification algorithm. This conventional approach is shown in Figure 4.1 (a).

Convolutional networks have been recently applied to classify remotely sensed images with very high resolution [110, 128, 95, 98, 113, 146, 164]. But, aside from [86] which used a combination of patch-based CNN and stacked autoencoders to fuse PAN and MS images, the majority of the works did not address the problem of multiresolution VHR images. A patch-based CNN [98] and a fully convolutional network (FCN) [113] were used to detect informal settlements from a pansharpened VHR image. Fully convolutional networks were also used to classify urban objects in VHR images both acquired in aerial and space-borne sensors [110, 128, 95, 146]. Moreover, [110, 128, 164] also utilized a separate post-classification step for map regularization. In this paper, we design a novel single-stage network performing image fusion, classification, and map regularization of a multiresolution VHR image in an end-to-end manner.

We propose a recurrent multiresolution convolutional network, called ReuseNet, to perform image fusion, classification, and map regularization of a multiresolution VHR image in an end-to-end fashion. We incorporate recurrence in the FuseNet base architecture to model contextual *label-to-label* dependencies and effectively regularize classification maps. We call this improved version ReuseNet. Contextual label dependencies are incorporated in ReuseNet by feeding classification scores of a previous FuseNet instance to a succeeding

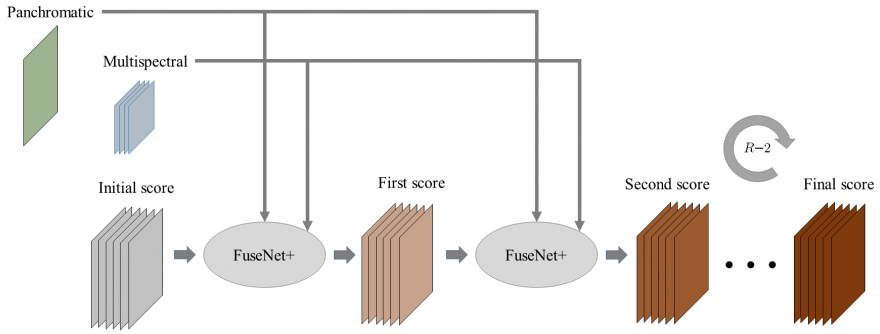


Figure 4.2 The general architecture of ReuseNet with R FuseNet+ instances. FuseNet+ employs exactly the same operations as FuseNet except for the first layer applying additional sets of convolutional filters on the input score maps. ReuseNet accepts three streams of input: 1) $\mathbf{x}_{PAN}$, 2) $\mathbf{x}_{MS}$, and 3) score maps of the same resolution as $\mathbf{x}_{PAN}$. It applies the same operations employed by a FuseNet in R cycles, taking the output score map of the previous cycle as an input.

one. Moreover, we introduce and compare a novel method to initialize the parameters and initial score maps of a ReuseNet.

4.2 Data and Methods

4.2.1 ReuseNet

ReuseNet builds on top of the architecture of FuseNet (see Chapter 3) by incorporating recurrent connections. Incorporation of this recurrent architecture in a full patch labeling approach enables the network to learn contextual label-to-label dependencies by feeding output score maps of a FuseNet instance to another instance of itself as an input. Such dependencies are similar to what graphical model (e.g. CRF/MRF) based methods learn in a post-classification regularization inference. For instance, a fully-connected CRF [75] solves an energy function that penalizes label configurations based on a unary term, often taken as the negative logarithm of the class scores [23], and a pairwise term, adding a penalization for pixels with different labels based on image-space and feature-space distances. Even though ReuseNet is based on the FuseNet architecture, the idea of learning contextual label information through recurrent

connections is independent of the choice of the base network architecture and can therefore be applied to other segmentation networks.

For ease of notation, let the series of operations performed by FuseNet (Equation 4.1) be given by the function f:

$$\mathbf{y} = f(\mathbf{x}_{PAN}, \mathbf{x}_{MS}) \tag{4.1}$$

where the $\mathbf{x}$'s are the input of FuseNet, and $\mathbf{y}$ is the class score map resulting from this input. The operations performed by ReuseNet are given by a recurrent variant g:

$$\mathbf{y}_1 = g(\mathbf{x}_{PAN} \oplus \mathbf{y}_0, \mathbf{x}_{MS}) \tag{4.2a}$$
$$\mathbf{y}_r = g(\mathbf{x}_{PAN} \oplus \mathbf{y}_{r-1}, \mathbf{x}_{MS}) \tag{4.2b}$$

where the r score map is obtained by applying the same function to a combination of the previous $r-1$ score map and the original FuseNet input as a new input. The recurrent variant g (denoted as FuseNet+ in Figure 4.2) applies exactly the same operations as f except for the first operation that instead of only taking $\mathbf{x}_{PAN}$ as an input, this operation takes the concatenation of $\mathbf{x}_{PAN}$ and a class score map $\mathbf{y}_r$ associated to the network instance r. Figure 4.2 shows a diagram illustrating the general architecture of ReuseNet.

We tested ReuseNet with several number of FuseNet instances (2, 3, and 4), calling each ReuseNet-R where R is the number of FuseNet instances within the ReuseNet. We also investigated various methods for initializing weights and initial score maps $\mathbf{y}_0$ of ReuseNet. Plain ReuseNet initializes the score maps with zeros, while ReuseNet$_{map-init}$ initializes the score maps using scores from a pre-trained FuseNet showing the best results in the fusion comparison experiments. We further extend ReuseNet$_{map-init}$ by initializing the weights of the FuseNet instance in the ReuseNet with the same FuseNet that provides the initial score maps. We call this extension ReuseNet$_{map-weights-init}$.

4.2.2 Perspective on Incorporating Recurrent Connections

The parameter sharing across FuseNet instances in a ReuseNet is consistent with the definition of a recurrent network, i.e. a recurrent network is a feedforward network that keeps on reusing the same set of weights to cycle through a sequence. The authors in [115] view such incorporation of recurrence as a way to increase the contextual window size, equivalent to the patch size M in a patch classification approach, of their patch classification based approach while controlling the capacity of the network via inter-instance weight sharing.

Table 4.1 Number of labeled pixels in each tile

Tile	Number of labeled pixels	Set
100	2178768	Training
105	2173602	Training
45	2063971	Validation
78	1977336	Test
82	1961955	Test

While both increase in contextual window size and capacity control of a CNN-based image patch classifier helps to improve the latter's performance, the first benefit is lost in a full patch labeling approach. In a fully convolutional network implementing full patch labeling, the contextual window size does not change as recurrent operations are added to the network since the contextual window size is equivalent to the effective size of the receptive field of the network. The effective size of the receptive field of the network depends on kernel sizes and strides of the network's convolutional and pooling operations, which are fixed and the same across instances.

In the proposed ReuseNet, recurrence integrates *contextual label information* to our model by considering class score maps as inputs to each FuseNet instance. This allows the model to learn label-to-label dependencies in addition to the spatial contextual information learned from the pixel values, *pixel-to-label* dependencies. This is a form of structured output prediction [7] where interdependencies between outputs may be expressed in terms of constraints restricting permissible output combinations or a more flexible form such as spatial dependencies across different output variables. Graphical models such as conditional random fields [78] are commonly used for such structured prediction tasks. ReuseNet uses operations in a deep convolutional network to learn features from both the input image and class scores—integrating the learning of label-to-label dependencies from the data instead of explicit image-space and feature-space distances as represented in a pairwise potential of CRF. This allows ReuseNet to be trained end-to-end as opposed to a two-stage approach applying a post-classification MRF/CRF as done in [46] and [128].

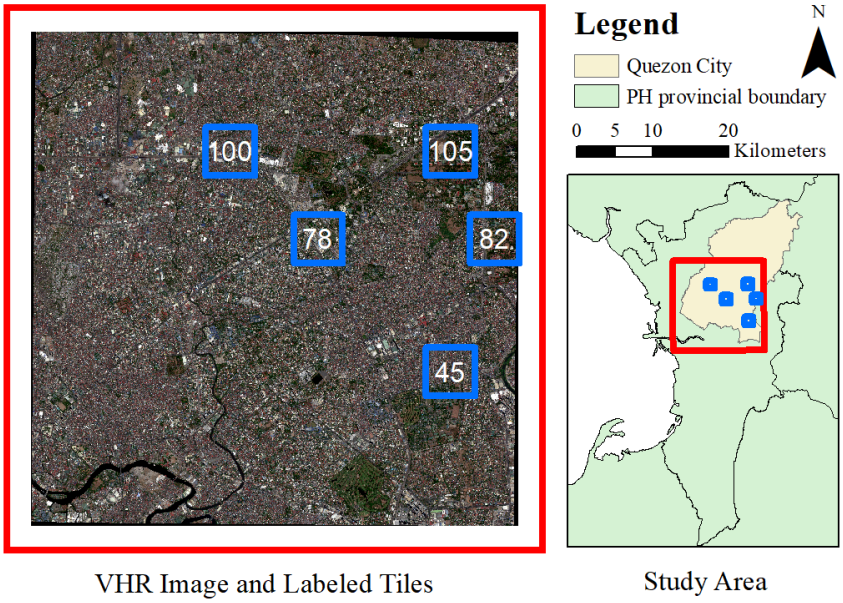


Figure 4.3 Figure showing the true color VHR image together with the locations of the labeled tiles (in blue squares) and the study area: Quezon City, Philippines.

4.2.3 Dataset Description

4.2.3.1 Worldview-03 Quezon City dataset

I evaluated the proposed networks in the land cover classification of a dataset covering Quezon City, Philippines. The dataset is composed of a Worldview-03 satellite image of the city acquired on 17^{th} April 2016 and corresponding manually prepared reference images for five chosen tiles (subsets) of the satellite image. The satellite image has a PAN band of 0.3 m resolution and four MS bands (near-infrared, red, green, and blue) of 1.2 m resolution. Reference images were prepared via photointerpretation and set to have the same spatial resolution as the PAN image. The whole satellite image was first divided into regularly-sized image tiles. PAN image tiles have a dimension of 3200 pixels $\times$ 3200 pixels, while MS image tiles have a dimension of 800 pixels $\times$ 800 pixels. Five non-adjacent tiles were sparsely labeled—annotating a pixel with a label belonging to one of the following six classes: impervious surface, building,

low vegetation, tree, car, and clutter. Two of the five labeled tiles were used for training (100 and 105), one for validation (45), and the remaining two for testing (78 and 82). Training samples are composed of pairs of image patches with dimensions $M \times M$ (taken from the MS image) and $4M \times 4M$ (taken from the PAN image tile). Figure 4.3 shows the VHR image and the corresponding locations of labeled tiles in the study area while Table 4.1 shows the number of labeled pixels in each image tile. Training samples were normalized to have a value between zero and one. The reference image patches have been converted into a "one-hot" encoding—a vector having zero values except for the index corresponding to the code of the class.

4.2.3.2 ISPRS Vaihingen dataset

For the ReuseNet experiments, we utilized the ISPRS 2D semantic labeling benchmark dataset of Vaihingen as an additional dataset [31]. We adopted the experimental setup used in [128, 146], employing the same training and validation tiles, to provide comparable results. We followed the sampling done in [128], except that data augmentation was not applied—resulting in less training samples. The method discussed in [103] was employed to extract the normalized DSM.

4.2.4 Comparison of methods

We compared ReuseNet against FuseNet using fully-connected CRF [75], denoted as FuseNet+CRF, to assess the capability of our classifiers to spatially regularize the classification results. The FuseNet+CRF baseline is similar to the approach adopted in [23, 128] but applied to PAN and MS images with different spatial resolutions. Spatial and feature space distances in the pairwise potentials of the fully-connected CRF are computed from the PAN image. We performed a grid-search of the CRF parameters, i.e. the weights and standard deviations of the appearance and smoothness kernels, and used the set of the parameters with the highest accuracy on the validation tile. We fixed the number of iterations to 10 for the mean field approximation algorithm used to perform inference in a fully-connected CRF.

We trained all the networks using a set of 17409 image patches taken from the training tiles and used 8255 image patches taken from the validation tile for early-stopping. We performed a random sampling with the constraint that the pixel near the center of the image patch is labeled. This may produce overlapping patches unlike the systematic gridwise sampling approach used in [128]. Gridwise sampling reduces the number of training patches since the

reference images is sparsely labeled, only around five percent of the pixels are labeled. The total loss value computed over a mini-batch is the total loss of all pixels divided by the number of labeled pixels within the mini-batch.

The FuseNets are trained using backpropagation with stochastic gradient descent setting the initial learning rate $\eta = 0.01$, momentum $\alpha = 0.9$, mini-batch size $N = 32$, and maximum number of epochs $\mathcal{T} = 240$. We decrease the learning rate in a stepwise manner as done in [61]—multiplying it by a factor of 0.1 after 60 and 180 epochs. The weights were initialized as in [47]. We did not find dropout to be helpful; hence, we only used an l_2-weight decay penalty—setting $\lambda = 0.001$—and a variant of early-stopping to regularize FuseNet. For early stopping, the classification accuracy on the validation set is calculated every epoch and the last model with the best validation accuracy is fixed to be the final instance of the model.

The FuseNet instances within a ReuseNet are identical, sharing the same network configuration and parameters. Each instance also couples a cross-entropy loss function with each of their score map. The total objective loss of a ReuseNet is the average of the cross-entropy loss values from all the FuseNet instances. We also used the same backpropagation with stochastic gradient descent setting as training a FuseNet with the initial learning rate $\eta = 0.01$, momentum $\alpha = 0.9$, mini-batch size $N = 32$, and maximum number of epochs $\mathcal{T} = 240$. Likewise, we decreased the learning rate in a stepwise manner—multiplying it by a factor of 0.1 after 60 and 180 epochs. For regularization, we only used an l_2-weight decay penalty—setting $\lambda = 0.001$. We can infer classification map from a ReuseNet in the same manner of inference as a FuseNet, with one additional option: to extract different predictions from each FuseNet instance.

For applying ReuseNet on the ISPRS Vaihingen dataset, we employed a feedforward network similar to the No-downsampling FCN proposed by [128] truncating the last two layers (fc5 and fc6) before softmax activation and entirely removing all maximum pooling without downsampling operations. With only convolutional layers (without pooling), we call this network AllConvNet. The network was trained on 12717 training patches as opposed to the 123330 training patches in [128]. Although having less parameters and having trained with a smaller number of training samples, AllConvNet provided comparable results with the original No-downsampling FCN while requiring less operations. We trained AllConvNet for 150000 iterations as reported in [128]. ReuseNet versions of AllConvNet were applied to the ISPRS Vaihingen dataset and were compared to the best results of both [128] and [146]. All the networks in this additional set of experiments were trained using a variant of SGD proposed in [157].

4.2.5 Accuracy Assessment

We compared the results of the different approaches using global measures: 1) overall classification accuracy (OA), 2) the Kappa coefficient (κ), 3) average class accuracy (AA), 4) and average class-F1 scores (F1). OA is given by:

$$OA = \frac{\sum_{i=1}^{C} n_{ii}}{n} \tag{4.3}$$

where n_{ii} is the number of samples classified as class i in both the the predictions and reference images, n is the total number of labeled samples in the reference images, and C is the number of classes, whereas κ is given by:

$$\kappa = \frac{n \sum_{i=1}^{C} n_{ii} - \sum_{i=1}^{C} n_{i+} n_{+i}}{n^2 - \sum_{i=1}^{C} n_{i+} n_{+i}} \tag{4.4}$$

where n_{i+} and n_{+i} are the number of samples classified as class i in the predictions and reference images respectively. Both OA and κ provides the rate of correctly classified pixels with the latter compensating for random agreement in classification. These global measures, however, are biased toward frequently occurring classes—meaning, classes with less frequencies have relatively little impact to the two measures. Unlike OA and κ, AA and F1 provides average of measures independent of class distribution. AA is given by:

$$AA = \frac{1}{C} \sum_{i=1}^{C} \frac{n_{ii}}{n_{i+}} \tag{4.5}$$

while F1 is given by:

$$F1 = \frac{1}{C} \sum_{i=1}^{C} \frac{2 \frac{n_{ii}}{n_{i+}} \frac{n_{ii}}{n_{+i}}}{\frac{n_{ii}}{n_{i+}} + \frac{n_{ii}}{n_{+i}}} \tag{4.6}$$

AA computes the average within-class rate of correctly classified pixels, while F1 calculates the harmonic mean of the precision (user's accuracy) and recall (producer's accuracy). We also observe and comment on the quality of the resulting classified maps.

Table 4.2 Comparison of map regularization approaches on Worldview-03 Quezon City dataset

Network	OA (%)	κ (%)	AA (%)	F1 (%)
FuseNet	91.90	88.43	93.46	81.74
FuseNet+CRF	93.07	90.08	**94.71**	81.72
ReuseNet-2	92.82	89.69	94.09	82.64
ReuseNet-3	92.98	89.88	94.54	85.42
ReuseNet-4	**93.49**	**90.58**	94.53	86.67
ReuseNet-5	92.74	89.53	92.78	**87.29**

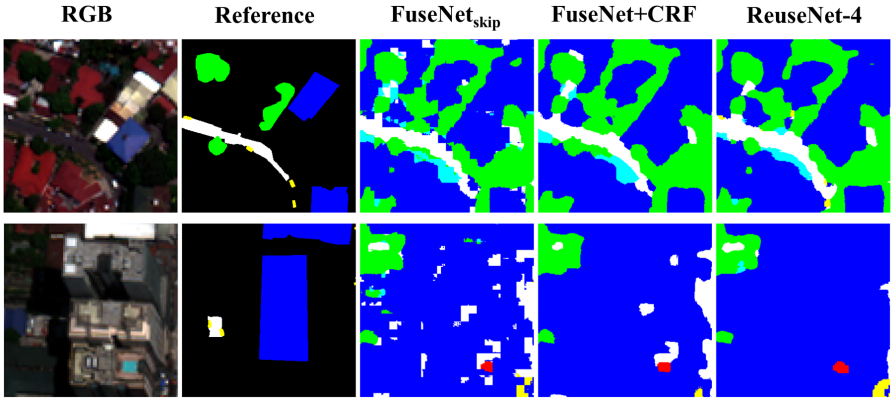


Figure 4.4 Two subsets from the test tiles showing, from right to left, the satellite image (natural color), reference image, and classification maps from FuseNet$_{skip}$, FuseNet$_{skip}$+CRF, and ReuseNet-4. All ReuseNets reported are "plain" meaning initial score maps are filled with zeros. Reusenet-R denotes a ReuseNet composed of R number of FuseNet instances.

4.3 Results and Discussion

4.3.1 Worldview-03 Quezon City dataset

Table 4.2 shows the accuracies obtained by comparing different classification techniques on the Worldview-03 Quezon City dataset. We found that both the ReuseNet instances and the baseline method FuseNet+CRF improves the numerical results of the plain FuseNet$_{skip}$ gaining around: 0.9–1.5% in OA, 1.2–2.1% in κ, and 0.6–1.2% in AA. For the F1, however, FuseNet+CRF

method performs worse than the plain FuseNet losing 0.02%; while all the other ReuseNet instances improves the F1 by around 0.9–5.5%. ReuseNet-4 outperforms all the other classifiers in all the metrics except for AA and F1—where both ReuseNet-3 and FuseNet+CRF outperform it by some margin in AA (0.01% and 0.18% respectively) and ReuseNet-5 considerably outperforms it in F1 by 0.62%. In particular, all the ReuseNets consistently show better F1 compared to both FuseNet and FuseNet+CRF—gaining almost 6%. These expected relatively smaller gains in numerical accuracy is consistent with what the author in [128] found—applying a post-classification CRF to an FCN to classify extremely high resolution aerial imagery increases the overall classification accuracy by around 0.1–1.0%. More noticeable changes are expected in the resulting improved regularity of the classified maps.

The numerical results above supports our assertion that introducing contextual label information through recurrence in an FCN applying a full-patch labeling approach can improve the classification of a VHR image. Such incorporation of label information allows our classifier to learn both pixel-to-label and label-to-label contextual dependencies. We can develop an intuition of these two dependencies by using an analogy to photointerpretation. We can easily imagine that it is easier to label a pixel when viewed with its neighboring pixels. This setup is analogous to the improvements a spatial-contextual classifier, like a CNN applying a patch classification, approach bring over a simple pixel-based classifier. But we can also see that it is easier to label a pixel when, aside from viewing its neighboring pixels, its surrounding pixels' labels are given. With contextual label information, the classifier can learn and leverage class spatial co-occurrences. Additionally, we observe that adding more FuseNet instances to the ReuseNet until $R = 4$ increases the score of all metrics, except for the average class accuracy where ReuseNet-3 marginally outperforms ReuseNet-4. Adding one more instance only improves the F1 score and degrades the other three metrics. We can interpret this addition of FuseNet instances as a way to increase ReuseNet's capacity to refine contextual label information fed to it as latter FuseNet instances receive more refined labels.

Figure 4.4 shows classification results of the best performing ReuseNet, the baseline method FuseNet+CRF, and the plain FuseNet. Both FuseNet+CRF and ReuseNet instances improves the quality of the resulting classified map by producing more regularized classification. We also observe that locations of the errors are carried over from the results of the FuseNet classifier from which both FuseNet+CRF and ReuseNet are based from. However, the occurrences of the errors are diminished especially on the facades of the large buildings. Detection of isolated cars in roads were also improved. Overall, results of ReuseNet-4 show better-quality classified maps by reducing noise in the classification

Table 4.3 Comparison of map regularization approaches on ISPRS Vaihingen dataset

Network	OA (%)	κ (%)	AA (%)	F1 (%)
NFCN [128]	87.17	–.–	–.–	–.–
CNN-FPL [146]	87.83	83.83	81.35	83.58
AllConvNet	86.98	82.71	87.17	85.46
NFCN with CRF [128]	87.90	–.–	–.–	–.–
ReuseNet-2	87.11	82.89	85.09	85.38
ReuseNet-3	**88.08**	**84.18**	**87.29**	**87.24**
ReuseNet-4	87.64	83.59	87.18	86.81

NFCN is a shorthand for No-downsampling fully convolutional network
Unreported values in the reference are denoted by "–.–"

(such as island of impossibly small buildings), further improving delineation of classes with irregular boundaries, and reducing misclassification in regions with ambiguous spectral characteristics such as facades and rooftops of high rise buildings.

4.3.2 ISPRS Vaihingen dataset

Table 4.3 shows the accuracies obtained by comparing different classification techniques on the ISPRS dataset. These results are in agreement with the results from the previous dataset. All the ReuseNet versions of AllConvNet improve the resuslts on all the four metrics except for AA and F1 of ReuseNet-2 (2.08% in AA and 0.06% in F1 respectively). ReuseNet-3, the best performing network, considerably improves all the numerical results of the plain AllConvNet by 1.1% in OA and is comparable and even greater than the 0.73% gain after a post-classification CRF in [128], 1.47% in κ, 0.12% in AA, and 1.78% in F1. ReuseNet-3 also outperforms best results reported in both [128] and [146].

These results reconfirm that introducing contextual label information through recurrence in an FCN applying a full-patch labeling approach can improve the classification of a VHR image. Similarly, qualitative improvements—such as holes in building being filled, better delineation of all classes in general, lesser artifacts—in the resulting classified maps are observed when ReuseNet is applied as shown in Figure 4.5.

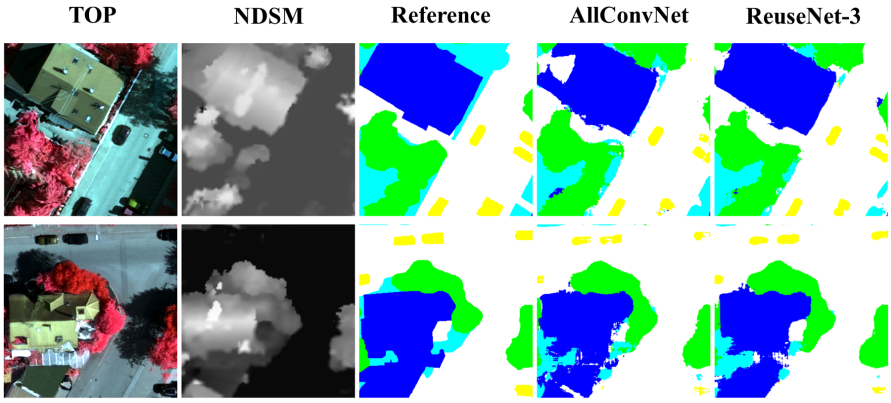


Figure 4.5 Two subsets from the validation tiles of ISPRS Vaihingen dataset showing, from right to left, the true orthophoto, normalized dsm, reference image, and classification maps from AllConvNet and ReuseNet-3 (best performing ReuseNet in this dataset). All ReuseNets reported are "plain" meaning initial score maps are filled with zeros. Reusenet-R denotes a ReuseNet composed of R number of FuseNet instances.

4.3.3 Different initializations

Figure 4.6 shows results of quantitive metrics on the three different ReuseNet initializations. There is low variation in the OA and κ. The trend of the two global scores is also inconsistent across the ReuseNet instances. For ReuseNet-2 and ReuseNet-3, the scores increases marginally (around 0.5% for OA and 0.8% for κ) when initialized with both the scores and weights from a previously-trained FuseNet. But for ReuseNet-4, there is a minor drop in both the scores (around 0.2% for both scores) when the two intialization methods are introduced. This could mean that increasing the FuseNet instances to a certain amount already provides enough room to a ReuseNet for "label refinement" such that gains from the initialization methods are compensated.

Introducing both initialization methods to a ReuseNet degrades the AA by around 0.9–5.2%. Applying only the initialization using scores from a FuseNet instance (map-init) degrades the F1 by around 0.9–12.2%. Interestingly, the F1 improve by around 0.8–6.1% when both initialization methods are introduced (map-weights-init). Decrease in AA can only imply an increase in false positive predictions in most of the classes; while increase in F1 could either mean decrease in false positive predictions or decrease in false negative predictions or both in most of the classes. The results therefore show that the initialization methods

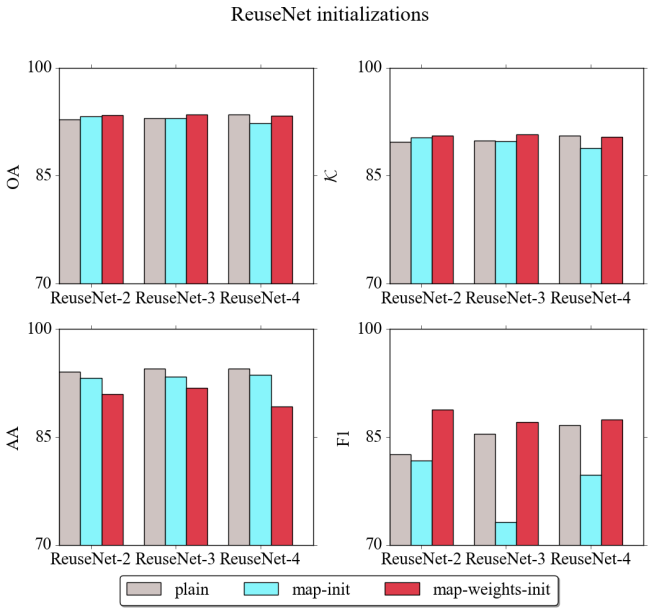


Figure 4.6 Plots showing results of quantitive metrics comparing different ReuseNet initializations; plain, map-init, and map-weights-init correspond to intializing the ReuseNet with zero-score maps, scores from a previously-trained FuseNet, and scores and weights from a previously-trained FuseNet respectively.

promote higher recall rate (decrease in false negatives) in underrepresented classes such as cars.

4.4 Conclusion

In this study, we presented a recurrent multiresolution convolutional network named ReuseNet to classfiy VHR satellite images. The operations for fusing the bands with different resolutions are learned within convolutional layers with corresponding downsampling and upsampling operations to match the resolution of the images. Regularization of the resulting classified maps is achieved by incorporating contextual label information through the recurrent architecture of ReuseNet. Additionally, we investigated various ways to initialize ReuseNet. The effect of varying a set of chosen network hyperparameters to

the classification accuracy of the network was explored. Both numerical and qualitative results show the advantages of incorporating image resolution matching and contextual label learning within the training of the classifier. To this end, we provided a single-stage classification pipeline incorporating image fusion, feature extraction, and map regularization, all combined in a convolutional network trained in an end-to-end manner. We designed the presented network architecture such that it can easily be adapted to other multiresolution image datasets. Inclusion and leverage of contextual label information is also separate from the design of the fusion network in the sense that it can be implemented on network classifying single-resolution images.

Urban Land Use Classification using Deep Multitask Networks

5

Abstract

Updated information on urban land use allows city planners and decision makers to conduct large scale monitoring of urban areas for sustainable urban growth. Remote sensing data and classification methods offer an efficient and reliable way to update such land use maps. Features extracted from land cover maps are helpful on performing a land use classification task. Such prior information can be embedded in the design of a deep learning based land use classifier by applying a multitask learning setup—simultaneously solving a land use and a land cover classification task. In this study, we explore a fully convolutional multitask network to classify urban land use from very high resolution (VHR) imagery. We experimented with three different setups of the fully convolutional network and compared it against a baseline random forest classifier. The first setup is a standard network only predicting the land use class of each pixel in the image. The second setup is a multitask network that concatenates the land use and land cover class labels in the same output layer of the network while the other setup accept as an input the land cover predictions, predicted by a subpart of the network, concatenated to the original input image patches. The two deep multitask networks outperform the other two classifiers by at least 30% in average F1-score.

This chapter is based on:
J. R. Bergado, C. Persello, and A. Stein. Land Use Classification using Deep Multitask Networks. XXIV ISPRS Congress 2020, Sep 2020 (accepted for publication).

5.1 Introduction

Urban land use maps provide essential information on the utilization of urban spaces. Updated information on urban land use allows city planners and decision makers to conduct large scale monitoring of urban areas for sustainable urban growth. Remote sensing data and methods offer an efficient and reliable way to update such land use maps. Using images regularly acquired by spaceborne and airborne sensors provide a much higher degree of objectivity and automation than traditional in-situ mapping methods. In this manner, extensive and updated information on urban land use can be made available on a regular basis.

Unlike land cover which describes the physical objects on a land surface, land use describes the human activity related to a specific geographical location. Since land cover is related to the physical properties of the land, the former is more straightforward to associate with observable physical properties that can be captured by sensors, such as land surface geometry and reflectance, than land use. Therefore, land use classification requires advanced classification techniques to be able to deliver functional land use maps.

Knowledge-driven rule sets from object-based classification techniques have been employed for such purpose [147]. However, those require tedious crafting of features extracted from the input data. More recently, deep learning techniques applied on remote sensing data further automated this feature crafting step by learning empirical data representations that are optimized for the classification task [12, 98, 65, 114, 161]. Particularly, [65] used a patch-based convolutional network combined with a post-classification trimming step to classify land use from high spatial resolution multispectral imagery; while [161] used an object based CNN to classify land use from very high resolution imagery.

A subset of handcrafted features employed in knowledge-driven land use classification can also be extracted from land cover maps [147]. Such prior information can be embedded in the design of a deep learning based land use classifier by applying a multitask learning setup—simultaneously solving a land use and a land cover classification task. Fully convolutional network variants have also been recently found to be more effective than their patch-based counterparts (Volpi and Tuia, 2017). In this study, we explore a fully convolutional multitask network to classify urban land use from very high resolution (VHR) imagery. To the best of our knowledge, this is the first study to explore performing a land use and a land cover classification simultaneously, in an end-to-end manner.

Figure 5.1 Sample image and corresponding reference for test Tile 1.

5.2 Data and Methods

In this study, we utilized a deep fully convolutional multitask network to perform urban land use classification from VHR imagery. The dataset comprises of a Worldview-03 satellite image of Quezon City, Philippines acquired on 17^{th} April 2016 and manually prepared reference images extracted by updating a Land Use Map of Metro Manila, the capital region where Quezon City is. Fully labeled reference images of land use classes were obtained from this step. Sparsely labeled reference images for the land cover classes to be used by the multitask networks were manually prepared via photointerpretation. The satellite image has a panchromatic band of 0.3 m resolution and four multispectral bands (near-infrared, red, green, and blue) of 1.2 m resolution.

The satellite image was pan-sharpened using the Gram-Schmidt pansharpening technique (Laben and Brower, 2000) and was subdivided into smaller non-overlapping image tiles of size 3200×3200 pixels. Twelve tiles were chosen, taking into account the presence of land use classes of interest, and grouped into training, validation, and testing set—six for training, three for validation,

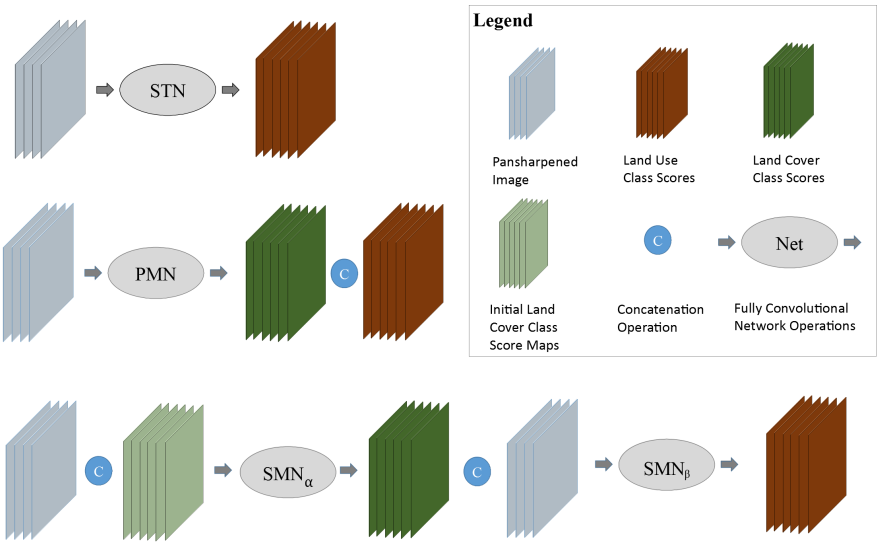


Figure 5.2 Illustration of the three different fully convolutional networks. STN (single task network) is a standard network only predicting the land use class of each pixel in the image. PMN (parallel multitask network) is a multitask network that concatenates the land use and land cover class labels in the same output layer of the network while SMN (sequential multitask network) accept as an input the land cover predictions, predicted by a subpart of the network, concatenated to the original input image patches.

and three for testing. Reference images corresponding to the 12 input image tiles were prepared with 6 land use classes: i) educational and cultural, ii) residential, iii) religious and cemetery, iv) informal settlements, v) commercial and industrial, vi) government and military. The image tiles were systematically sampled into smaller non-overlapping 128×128 image patches that are then fed as input to the network. The training set was further augmented by two flips and three 90°rotation transformations. Figure 5.1 shows a sample image and reference tile from the test set.

5.2.1 Standard approach

We used two methods as baseline approaches to be compared to our proposed methods. Firstly, a pixel-based random forest classifier trained to classify land use from the input pansharpened images. Secondly, a standard fully convolu-

tional network (FCN) classifying land use from the same input pansharpened images; but instead of accepting a 1D input vector of pixel values as done in the random forest classifier, accepts a 3D array of values from the 128×128 image patches, and thus, takes spatial context into account. For notational purposes, we call this network STN (single task network).

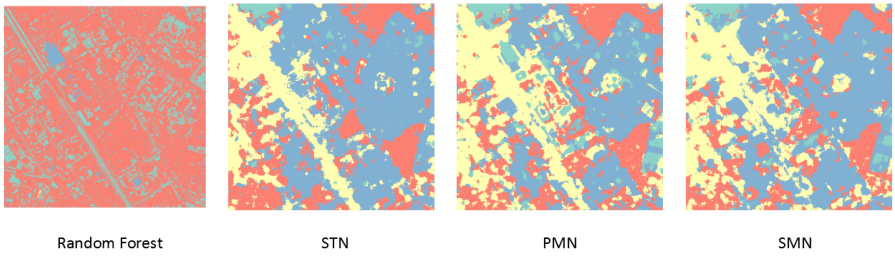


Figure 5.3 Predicted land use maps of the four classifiers on test Tile 1.

We used a modified version of U-Net [118] as the base network architecture of all our convolutional networks. The weights of the encoder is also initialized from a pretrained VGG16 (Simonyan and Zisserman, 2015). The modification, similarly done by [128], involves adding an additional set of kernels in the first convolutional layer, this additional kernel is initialized from randomly choosing one of the three original kernels from the pretrained VGG16. The number of channels of the output layer was also correspondingly changed to be equal to the number of our target land use and land cover classes.

5.2.2 Proposed approach

The proposed approach multitask LULC networks has two variants: first is a multitask network that concatenates the land use and land cover class labels in the same output layer of the network; second is a variant that accept as an input the land cover predictions, predicted by a subpart of the network, concatenated to the original input image patches. We call the first variant PMN (parallel multitask network) and the second one SMN (sequential multitask network). The three networks can be represented by the following functions:

$$y_u = \text{STN}(x) \tag{5.1a}$$

$$[y_u, y_c] = \text{PMN}(x) \tag{5.1b}$$

$$\begin{cases} y_c = \text{SMN}_\alpha([x, y_c^0]) & \text{(5.1c)} \\ y_u = \text{SMN}_\beta([x, y_c]) & \text{(5.1d)} \end{cases}$$

where x is the pansharpened input image patch, y_u is the output land use predictions, y_c is the output land cover predictions, SMN_α and SMN_β are two sub networks of SMN, y_c^0 is land cover class initialization (in the experiments we initialized all the values to zero), and [ ] is a channel-wise concatenation operation. Equations 5.1c and 5.1d are jointly optimized. Figure 5.2 shows a diagram of the three networks, highlighting the difference between the input and output layers of each network.

All the networks were trained to optimize a cross-entropy loss:

$$E_N = -\sum_{n=1}^{N} \mathbf{t}_n \bullet \log(\mathbf{y}_n) \tag{5.2}$$

where E is the loss function value evaluated over N samples, $\mathbf{t}_n$ is a binary vector encoding the the target class labels (with the index corrresponding to a class having a value of 1 and 0 otherwise), $\bullet$ denotes the dot product, and $\mathbf{y}_n$ is the class score maps of a sample n calculated using a *softmax* activation function. STN and PMN defines one cross-entropy loss function in their output layers while SMN decomposes the total loss function into two equally-weighted cross-entropy losses at the output layers of SMN_α and SMN_β.

The loss was optimized using Adam (Kingma and Ba, 2014) for 150 epochs utilizing a batch size of 64. The base learning rate used was 0.0001 which was reduced by a factor of 10 every 50 epochs.

Table 5.1 Land use class frequency averaged over the whole set of image tiles

Class	Frequency (%)
Educational and Cultural	22
Residential	39
Religious and Cemetery	2
Informal Settlements	3
Commercial and Industrial	19
Government and Military	15

Since there is an imbalance in the distribution of the land use classes present in our image tiles (see Table 5.1), we assessed the classification performance of the four classifiers using the average class F1-score. This metric will be more robust to the class frequency imbalance than the standard overall classification

accuracy, the latter generally giving overly optimistic estimates of the classifier performance.

5.3 Results and Discussion

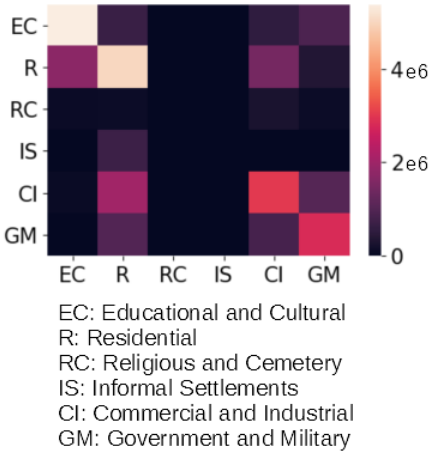


Figure 5.4 Confusion matrix of PMN on the three test tiles.

Table 5.2 Average land use class F1 scores of the classifiers on the three test tiles

Classifier	Tile 1	Tile 2	Tile 3
random forest	1.07	14.27	12.36
STN	25.48	21.59	29.63
PMN	57.90	**52.89**	**57.44**
SMN	**59.87**	52.53	53.34

The land use classification accuracy of the different classifiers assessed on the three test image tiles (1, 2, and 3) are shown in Table 5.2. PMN achieves the highest average class F1 score in two of the three test tiles, with SMN having better results for Tile 1. There is a considerable increase in the classification accuracy of the classifier by using a standard fully convolutional network over the baseline random forest classifier. This shows that the features learned

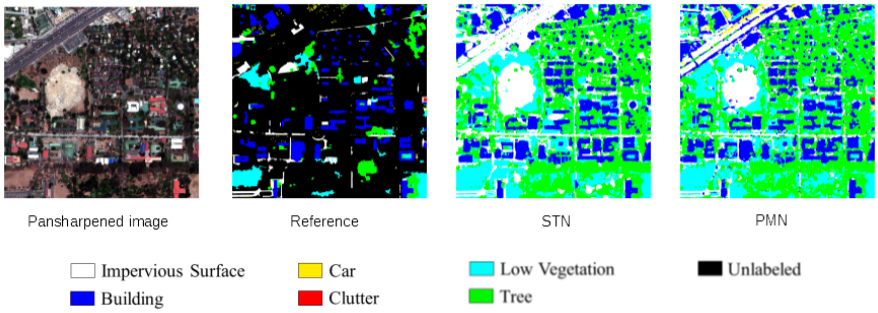


Figure 5.5 Comparison of predicted land cover maps of Tile 3 using the plain network STN and one of the multitask network PMN.

in the hidden layers of the convolutional network is helpful for this land use classification task. There is also an observable increase in performance, at least about 30% in average F1-score, when using the two multitask network over the standard one. Such improvements are consistent with the intuition that features extracted from land cover can help the land use classification task.

Figure 5.3 shows the predicted maps of all the classifiers on Tile 1. All the three networks produced maps of better quality than the baseline random forest classifier. All of three confuses the underrepresented (see Table 5.1) informal settlement classes as residential areas. This is due to the visual similarity of high density residential areas in the city to informal settlements. This is further affected by the limited number of training labels for this class. This can also be observed in the resulting confusion matrix (see Figure 5.4) of PMN on the three test tiles where it can clearly be seen the poor performance of the classifier on both the underrepresented two classes (religious and cemetery and informal settlements). On the other hand, the educational and cultural land use class appears to have the least misclassification compared to other classes.

Figure 5.5 shows a comparison of predicted land cover maps of Tile 3 using the plain network STN and one of the multitask network PMN. The plain network poorly classifies the car class which are confused with building pixels. Predictions of the underrepresented car class were greatly improved by using the multitask learning setup. There is also less overclassification of the impervious surface class after using the multitask network PMN. This shows that the learned features shared by both tasks help on improving the each other's predictions.

5.4 Conclusion

Classification of urban land use maps is essential to provide updated information on utilization of urban spaces. This study shows that performing land use classification simultaneously with classifying land cover improves the resulting classified land use maps. Comparing two multitask networks, we obtained an improvement of at least 30% in the average F1-score as compared to standard classification approaches. Such an approach can be embedded in the design of a deep learning based classifier. The multitask network also improves the predictions on the additionally embedded land cover prediction task.

Predicting Wildfire Burns from Big Geodata using Deep Learning

6

Abstract

Wildfire continues to be a major environmental problem in the world. To help land and fire management agencies manage and mitigate wildfire-related risks, we need to develop tools for mapping those risks. Big geodata—in the form of remotely sensed images, ground-based sensor observations, and topographical datasets—can help us characterize the dynamics of wildfire related events. In this study, we design a deep fully convolutional network, called AllConvNet, to produce daily maps of the probability of a wildfire burn over the next 7 days. We applied it to burns in Victoria, Australia for the period of 2006–2017. Fifteen factors that were extracted from six different datasets and resulted into 29 quantitative features, were selected as input to the network. We compared it with three baseline methods: SegNet, multilayer perceptron, and logistic regression. AllConvNet outperforms the other three baseline methods in four of the six quantitative metrics considered. AllConvNet and SegNet provide smoother and more regularized predicted maps, with SegNet providing greater sensitivity in discriminating less wildfire-prone locations. Input feature statistical importance was measured for all the networks and compared against logistic regression coefficients. Total precipitation, lightning flash density, and land surface temperature occur to be consistently highly weighted by all models while terrain aspect components, wind direction components, certain land cover classes (such as crop field and woodland), and distance from power lines are ranked

This chapter is based on:
J. R. Bergado, C. Persello, K. Reinke, and A. Stein. Predicting Wildfire Burns from Big Geodata using Deep Learning (submitted for review).

on the lower end. We conclude that wild-fire burn prediction methods based on deep learning present quantitative and qualitative gains.

6.1 Introduction

Wildfire continues to be a major environmental problem in the world [154, 19]. When poorly managed, it can cause permanent and undesirable changes to certain landscape and ecosystem services [22]. Although most incidents are found in remote areas and naturally occur as part of terrestrial ecosystem processes, human populations and infrastructure within wildland-urban interfaces are still exposed to wildfire related risks [101]. In the context of wildfire management, we consider wildfire risk as the probability of a wildfire event that may consequently result in expected loss of lives and degradation of assets. We can characterize such risk by mapping associated risk conditions. In general, these conditions are both spatially and temporally dynamic in nature and depend on a number of factors including: fuel conditions, meteorological variables, topography, and sources of ignitions [27]. Big geodata—in the form of multitemporal remotely sensed images, ground-based sensor observations, and topographical datasets—is a promising source of information on these wildfire risk factors.

Quantitative assessment of wildfire risk employs a four-stage framework [38, 139]. Firstly, problem formulation defines specific management objectives to be addressed. Secondly, exposure analysis quantifies the likely magnitude and spatiotemporal connection between the identified risk variables [38]. Thirdly, effect analysis quantifies the response of the valuable resources at risk as a function of the fire behavior—usually flame length [140] in combination with rate of spread and fire intensity. Finally, risk characterization integrates information from the three previous stages to come up with a complete, informative, and useful conclusion for decision-making [129].

Exposure analysis describes how likely will valuable resources of interest interact with a wildfire [140]. Thus, it directly deals with quantifying probabilities of wildfire events [1]. Probability of wildfire burn refers to how likely a geographical location will change from an unburnt state to a burnt state within a given time period, e.g. annually [100]. Estimates of this probability can either serve as a proxy measure for wildfire risk or as input to probabilistic methods to quantify wildfire risk [42]. Maps showing probability of wildfire burn can be further intersected with the asset locations [9, 3]—to identify likely affected assets.

Formal treatment to quantify the probabilities associated to wildfire events can become complicated. Most case studies on wildfire occurrence do not adopt a consistent quantitative definition. They employ a logistic regression (LR) [33, 20, 28, 5, 72, 55, 32, 111] trained on a number of relevant wildfire risk factors, acquired through remotely sensed data and data products, and information on historical wildfire locations. Within this group of studies, differences exist in interpretation. For example, Chuvieco et al. (2010) predicts the number of fire incidence reclassified into low and high incidence while Badia et al. (2011) reclassifies the output of the logistic function into five ranges. Other studies have applied methods such as multilayer perceptron (MLP) [33], weights-of-evidence Bayesian model [69], evidential belief function [108], and conversion from reference fuel moisture tables [124].

A wildfire event prediction problem can be seen as a function learning task, where the function maps wildfire-related input variables into probabilities of a wildfire event. Higher-level spatial and temporal association between the input variables may improve the predictive accuracy of the learned function. However, just like the mapping function, there is a knowledge gap on how to construct these higher-level features. Models based on deep learning can be used to capture such higher-level spatial and temporal association together with learning the mapping function.

There is limited work on employing deep neural networks and remotely sensed data for wildfire mapping. A recent relevant work presents a two-stage strategy on estimating weekly wildfire hotspots in Australia [37] using deep belief networks for feature compression before feeding the latter to an ensemble classifier. Most relevant works produce static probability maps estimated over a certain period of time [33, 20, 5, 32]. Although, wildfire-driving factors are dynamic in nature, hence, producing time-series maps, either sub-hourly [151], daily [28, 55] or weekly [37, 53] presents a better understanding of wildfire risk. Remotely sensed images and stationary ground-based sensors are promising sources of time-series information that can be used to produce these maps. We can combine these time-series information with historical wildfire burn records in a predictive model, e.g. a FCN, to produce dynamic probability maps of wildfire burn.

This research aims to utilize supervised deep learning techniques to estimate probabilities of wildfire burn by combining information from remotely sensed images, stationary ground-based sensors, topographical datasets, and historical wildfire data. Specifically, fully convolutional networks (FCN) are employed to produce daily predictions mapping probability of wildfire burns in Victoria, Australia for the period 2006–2017.

The major contributions of this study are:

- to design and implement a fully convolutional network for predicting daily maps of the probability of a wildfire burn over the next 7 days utilizing an extensive set of wildfire related input variables taken from various data sources such as: time series of satellite images and data products, climatological sensor observations, topographical geospatial databases, and historical wildfire burn records for Victoria, Australia over the period of 2006–2017;
- to quantitatively and qualitatively evaluate the proposed network against several baseline methods;
- and to quantify the relative statistical importance of each input features used in the deep fully convolutional network.

6.2 Data and Methods

6.2.1 Study area

Victoria (37°S 144°E) is the southeastern state of Australia. It has an area of approximately 238000 km^2, of which roughly 16% is forest, 10% is woodland, 5% is shrubland, and 6% is grassland. Wildfire is a natural component, making the state one of the most wildfire-prone areas in the world [138].

The spatial extent of wildfire burns within Victoria was extracted from the "Fire History Records of Fires primarily on Public Land" dataset made publicly available by the State of Victoria, Department of Environment, Land, Water & Planning (DELWP) [The State of Victoria, 1992]. This allows prescribed burns to be distinguished from wildfires, whereas the MODIS active fires dataset does not allow for this distinction and may miss fire activity with smaller burn extents [56] or lower intensities. Temporally, the Fire History Records dataset only provides the starting date of the fires. Figure 6.1 shows a map of the study area. Planned burns conducted by DELWP were filtered out and the remaining wildfire burn extents for the period of 2006–2017 were used in this study.

6.2.2 Input variables

The probability of a wildfire burning a specific geographical location depends on both natural and anthropogenic factors. The spatial and temporal scale of the study required the organization of a big geodataset, not only in terms of volume and update frequency but also data heterogeneity, matching wildfire risk factors with recorded wildfire extent locations. Fifteen factors that were

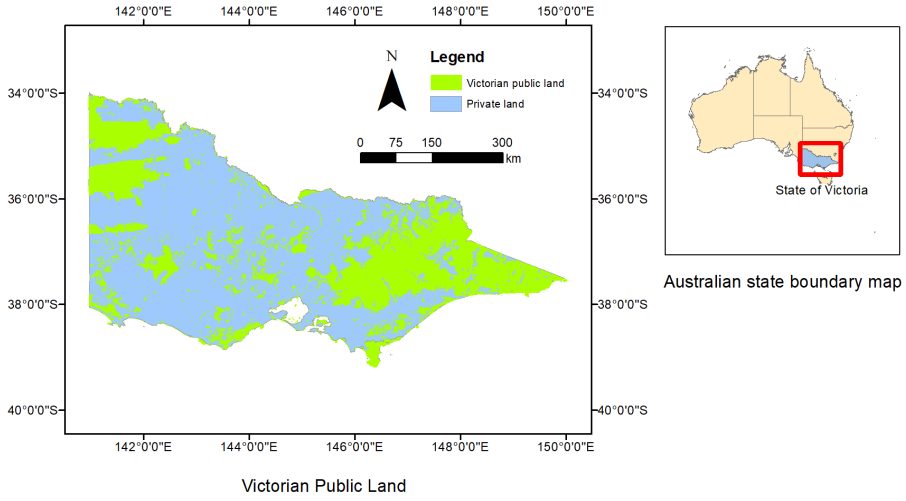


Figure 6.1 Location of the state of Victoria and its public land

extracted from six different datasets and resulted into 29 quantitative features, were selected as input to our models predicting probability of wildfire burn. These features encode the factors associated to wildfire burn such as topography (elevation, slope, and aspect), weather (temperature, humidity, solar radiation, rainfall, wind speed and direction, and lightning flash density), proximity to anthropogenic interfaces (distance to roads, distance to power lines) and fuel characteristics (fuel type, fuel moisture, emissivity). Table 6.1 shows these input variables including the dataset from which they are taken from and their corresponding spatial and temporal resolution before any spatial resampling.

We first downsampled the digital elevation model [Hutchinson et al., 2008] to 500 m resolution using bilinear interpolation to produce the elevation, slope, and the cosine and sine components of the aspect. Road and power line network vectors [The State of Victoria, 2009a, 2009b] were rasterized to a spatial resolution of 500 m. Euclidean distances from both the roads and the power lines resulted into two additional features extracted from this topographical vector database. These six features are temporally stationary.

We obtained land cover/land use information from the National Dynamic Land Cover Dataset of Australia [Lymburner et al., 2011]. The original dataset with a spatial resolution of 250 m and 22 land cover classes was downsampled using nearest neighbor into a spatial resolution of 500 m. The original 22 classes

were collapsed into eight (Table 6.2) based on similar land use, land cover and vegetation characteristics. The new land cover maps were transformed into binary presence maps, encoding the value one when a class is present and the value zero otherwise. This resulted into eight features extracted from the land cover dataset. The original dataset is obtained from classifying MODIS time series images spanning over two-year periods. The eight resulting features were temporally matched to other features using the year of the latest image from the two-year averaging period.

Table 6.1 Input variables associated to wildfire burns of Victoria, 2006–2017

Source Dataset	Input Variable	Units	Spatial Resolution	Temporal Resolution
Digital elevation model[a]	elevation	m	250 m	stationary
	aspect	—		
	slope	—		
Topographical databases[b]	proximity measures	m	—	stationary
Land cover/use[a]	surrogate for fuel type	—	250 m	bi-annual
MODIS images[c]	fuel moisture proxy	—	500 m	8 days
	land surface temperature	K		
	emissivity	—		
LIS data product[d]	lightning flash density	flashes·km^{-2}	12 km	DOY
Meteorological data[e]	temperature	°C	—	daily
	relative humidity	%		
	solar radiation	MJm^{-2}		
	precipitation	mm	—	1 minute
	wind speed	kmh^{-1}		
	wind direction	degrees		

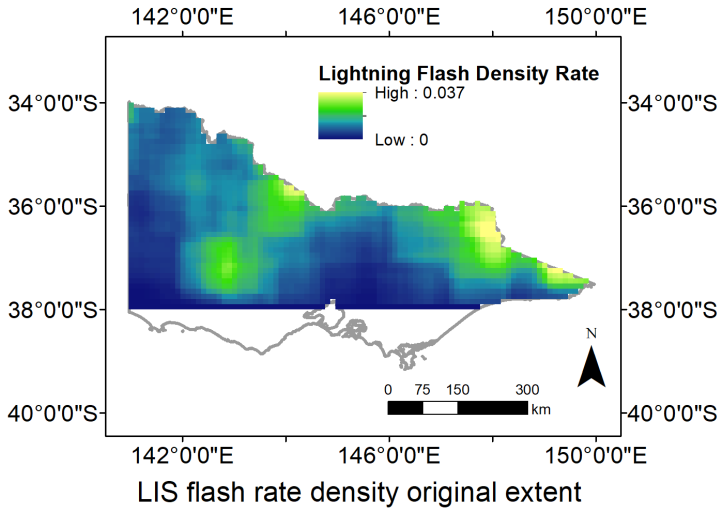
[a] Obtained from Geoscience Australia web portal [Hutchinson et al., 2008, Lymburner et al., 2011]
[b] Obtained from Victoria's open data directory [The State of Victoria, 2009a, 2009b];
[c] Obtained from NASA EOSDIS Land Processes DAAC data portal [Vermote, 2015, Wan et al., 2015];
[d] Obtained from NASA GHRC DAAC data portal [Cecil et al., 2014];
[e] Sourced from Australian Bureau of Meteorology;
DOY means observations are averaged based on the day of the year

Table 6.2 Aggregated land cover/use classes of the Australian Dynamic Land Cover Dataset

New Class ID	Original Classes
1	lakes and dams
	salt lakes
2	mines and quarries
	urban areas
3	irrigated cropping
	rain fed cropping
	irrigated pasture
	rain fed pasture
	irrigated sugar
	rain fed sugar
	wetlands
4	alpine meadows
5	open hummock grassland
	closed tussock grassland
	open tussock grassland
6	scattered shrubs and grasses
	dense shrubland
	open shrubland
7	closed forest
	open forest
8	woodland
	open woodland

A proxy for fuel moisture content was obtained from spectral indices derived from a time series of MODIS images [Vermote, 2015]. The images have a temporal resolution of eight days and a spatial resolution of 500 m. Four spectral indices correlated with fuel moisture content [18, 153] were used: the normalized difference vegetation index (NDVI), visible atmospherically resistant index (VARI), normalized difference infrared index (NDII), and the normalized difference water index (NDWI). One thermal band and two emissivity bands were directly used as input features [Wan et al., 2015]. This resulted into seven features extracted from MODIS images. The features are temporally matched using the latest date of the eight-day averaging period prior to the starting date of the wildfire.

A major natural cause of wildfire ignitions is lightning. Lightning flash density information was obtained from annual lightning climatology derived



LIS flash rate density original extent

Figure 6.2 LIS flash rate density original extent

from NASA's lightning imaging sensor (LIS) observations [Cecil et al., 2014]. The annual lightning climatology provided estimates of lightning flash density for each day of the year averaged over more than a 16-year period with a spatial resolution of about 12 km. The dataset is spatially incomplete beyond the south of the 38°S parallel (Figure 6.2). We therefore used direct sampling [96] to impute locations within the study area extent with missing lightning flash density rate. The imputed data were manually upsampled to 500 m spatial resolution using bilinear interpolation. This resulted into one feature that is temporally matched using the day of year (DOY) associated to the average lightning flash density rate image.

Finally, we selected five weather variables associated with wildfire [143, 122, 5, 55, 105, 99] namely: temperature, humidity, solar radiation, wind speed and direction, and precipitation. The first three have a daily resolution, whereas the last two have a minutely resolution. We selected the temperature as the maximum temperature past 9 am in 24 hours. For humidity, the lowest relative humidity from every three hours within a day was chosen. The solar radiation records the total amount of daily global solar exposure. Both the wind speed and direction and precipitation were aggregated into daily values—taking the average wind speed and direction and total precipitation within the day. Observations having bad quality indicators were removed and four days (of

interest) without any qualified solar radiation observations were temporally imputed using the average of the same DOY from years with measurements having good quality indicator. The five variables were interpolated using ordinary kriging into a spatial resolution of 500 m. This resulted into five features matched using the date associated to the meteorological observation.

The raster grids were matched to have the same geographic coordinate system (WGS 84) and spatial resolution of 500 m. The organized dataset resulted into 29-band input images and one-band target images of size 1536 × 2304 pixels for all dates between 2006 and 2017 (Figure 6.3). Figure 6.4 shows the location of the wildfire burns for the whole period. Directly feeding these input and target images to our predictive model would both severely limit the size of the convolutional neural network we can train and would make the learning process computationally slower (because of the spatial dimensions of the images) and difficult (considerable imbalance in the number of labels). To address such computational constraints, we sampled this images into smaller input patches.

6.2.3 Predictive model

In the context of wildfire prediction, the inputs of the artificial neural networks are equivalent to the independent variables associated to a wildfire event. Identifying the outputs of the network can be challenging. Specially since there has not been any consistent formal definition of probability of burn in the literature. A generalization of common neural network operations can be found in [13].

Feedforward artificial neural networks are often employed as deterministic models. But, the outputs of these networks can be treated with probabilistic meaning. For example, in a regression problem, we can assume that the target variable t follows a Gaussian distribution dependent on the input $\mathbf{x}$ and parameters $\mathbf{w}$ of the network f such that

$$p(t|\mathbf{x}, \mathbf{w}) = \mathcal{N}(t|f(\mathbf{x}, \mathbf{w}), \sigma^2) \tag{6.1}$$

where σ is the variance of the Gaussian [15, pp. 232–236]. The choice of the probability distribution is not constrained by the network itself but is problem-dependent, reflecting our expected distribution of the target variable t.

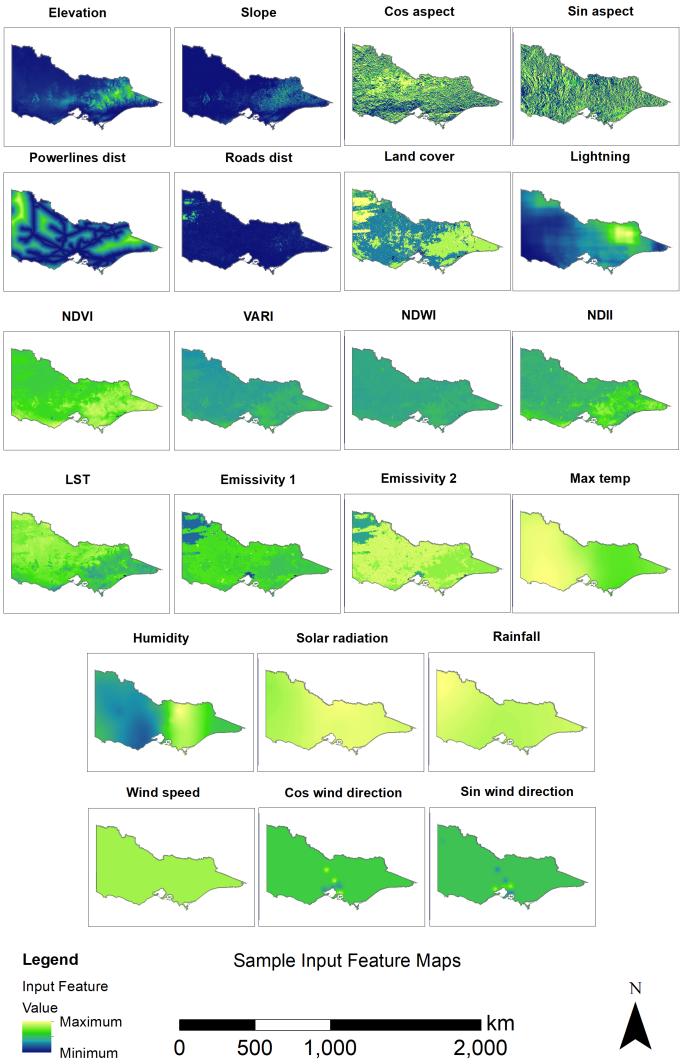


Figure 6.3 Visualization of the input images for one selected date (December 1, 2006). All images are continuous variables normalized to [0, 1], except for the fuel type map that shows discrete class categories from Table 6.2.

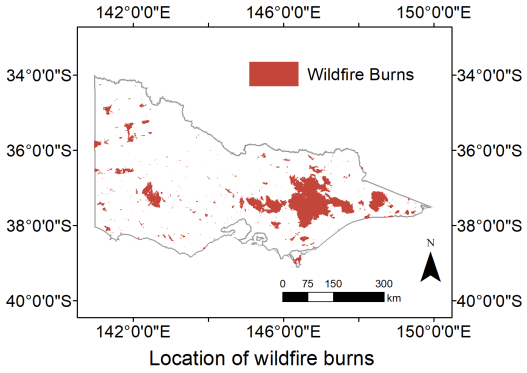


Figure 6.4 Visualization of the location of wildfire burns for the whole study period. Locations were obtained from the "Fire History Records of Fires primarily on Public Land" dataset made publicly available by the State of Victoria, Department of Environment, Land, Water & Planning (DELWP) [The State of Victoria, 1992].

6.2.4 Sampling of burn and non-burn locations

We systematically extracted non-overlapping sample patches of size $M \times M$, with the pixel near the center of the patch lying on a location of a wildfire burn, from both the wildfire burn location (target patches) and the 29 quantitative features (input patches) accordingly matched with the recorded starting date of a wildfire. Corresponding burn locations are temporally binned together in 7-day temporal windows, hence the resulting probability is an estimate of the likelihood of a wildfire burning a specific location within the next 7 days. All pixels in the target patches encodes a binary label, one for wildfire burn locations and zero otherwise, except for pixels lying outside the study area. Additionally, patches only having non-burn locations were randomly collected from dates with and without any recorded wildfire.

Each of the input features was normalized to a value of $[0, 1]$. Training patches were sampled from images within the period 2006–2016, while all test patches are taken from an entirely separate period 2017. Data augmentation, including two flips and three 90° rotations, was applied to increase the number of training patches and help prevent the networks from overfitting.

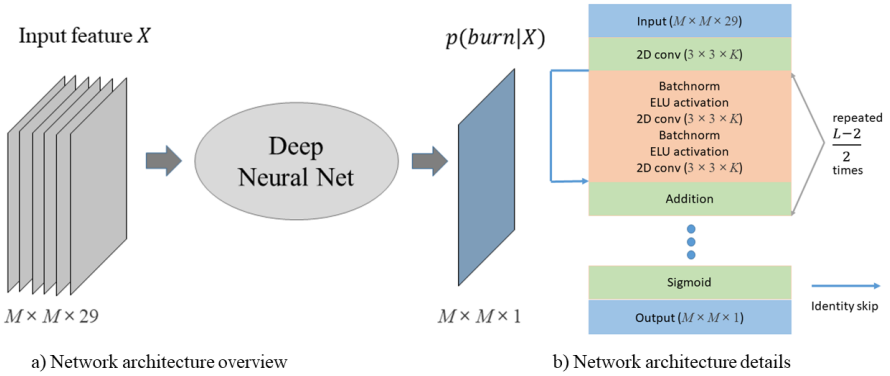


Figure 6.5 Simplified illustration of the deep fully convolutional network architecture (AllConvNet) we designed for predicting the probability of wildfire burn. Figure 5.a shows the main components of an artificial neural network—input layer, hidden layer, and ouput layer—specifying the dimensions of the first and last layers. Figure 5.b further shows the pattern of operations applied within the hidden layer of the network (a more detailed discussion of these operations can be found in [13]). The "2D conv" learns sets of 2D convolutional filters to extract higher-level features from the input, Batchnorm learns parameters used to normalize the values of the learned features to have zero mean and unit variance [68], ELU activation applies an element-wise non-linear activation function called exponential linear units [29], and Addition performs an element-wise addition.

6.2.5 Network architecture and learning setup

For predicting the probability of wildfire burn from our input and target patches, we designed a variant of deep fully convolutional network [87] employing residual blocks with identity connections [61]. Figure 6.5 shows a simplified diagram of the network architecture used in this study. For notational purposes, we call this network AllConvNet. In contrast to the original fully convolutional network, which uses both convolutional and maximum pooling with downsampling layers, AllConvNet only uses convolutional layers (hence the shortened name for all-convolutional network). We performed sensitivity analysis experiments on some chosen hyperparemters of the network including: the number of layers, number of filters, and input patch size of the network. For the last layer, the network applies a sigmoid activation function followed by a binary cross-entropy loss

function:

$$E_N(\mathbf{w}) = -\frac{1}{N}\sum_{n=1}^{N} h(\mathbf{t}_n)\mathbf{t}_n \log(\mathbf{y}_n) \tag{6.2a}$$

$$\mathbf{y}_n = f(\mathbf{x}_n) \tag{6.2b}$$

$$h(\mathbf{t}_n) = \begin{cases} 1 & \text{if } \mathbf{t}_n = 0 \\ \gamma & \text{if } \mathbf{t}_n = 1 \end{cases} \tag{6.2c}$$

where E is the loss function value evaluated over a batch size N, h is a class-based weighting function, γ is the class-weighting value (giving more importance to the positive samples, encoded as $\mathbf{t}_n = 1$), $\mathbf{t}$ is the target output vector, $\mathbf{y}$ is the model output vector, $\mathbf{x}$ is the input feature vector, and f is the series of operations performed by the network. The network is optimized using a variation of the backpropagation with stochastic gradient descent algorithm called "Adam" [74]. The networks are trained for 500 epochs and early stopping was applied by using the model with the best loss function value evaluated on a separate validation set taken from the original set of training patches.

6.2.5.1 Accuracy assessment

We compared three other methods against our network. Firstly, an adapted version of SegNet [6], an encoder-decoder FCN that has been mainly used and employed in pixel-wise image classification originating from computer vision but has also been employed in remote sensing applications [114]. Secondly, a multilayer perceptron. And finally, a logistic regression. Our network and SegNet accepts $M \times M$ input patches with 29 feature bands, while the MLP and LR only accepts a 1D input vector with 29 features. Both the other two artificial neural networks (SegNet and MLP) used the same optimization method as AllConvNet. We also used similar hyperparameter selection experimental setups and class-weighting function for all the three networks.

The labels of the target image predominantly consists of negatively-labeled (non-wildfire burn location) pixels. Hence, using pixel-wise overall classification accuracy would be inappropriate as predicting negatives for the whole output would still yield a high accuracy. For such cases of imbalance distribution of labels, more fitting accuracy metrics would be a recall-biased F-beta score

averaged over the two classes:

$$F_\beta = 0.5(F_\beta^+ + F_\beta^-) \tag{6.3a}$$

$$F_\beta^+ = \frac{(1+\beta^2)\mathrm{TP}}{(1+\beta^2)\mathrm{TP} + \beta^2\mathrm{FN} + \mathrm{FP}} \tag{6.3b}$$

$$F_\beta^- = \frac{(1+\beta^{-2})\mathrm{TN}}{(1+\beta^{-2})\mathrm{TN} + \beta^{-2}\mathrm{FP} + \mathrm{FN}}, \tag{6.3c}$$

the class balance accuracy [102] defined as:

$$CBA = 0.5(CBA^+ + CBA^-) \tag{6.4a}$$

$$CBA^+ = \frac{\mathrm{TP}}{\max(\mathrm{TP}+\mathrm{FP}, \mathrm{TP}+\mathrm{FN})} \tag{6.4b}$$

$$CBA^- = \frac{\mathrm{TN}}{\max(\mathrm{TN}+\mathrm{FN}, \mathrm{TN}+\mathrm{FP})}, \tag{6.4c}$$

and the Matthews correlation coefficient:

$$MCC = \frac{\frac{\mathrm{TP}}{N} - S \times P}{\sqrt{(P \times S)(1-S)(1-P)}} \tag{6.5a}$$

$$N = \mathrm{TP} + \mathrm{TN} + \mathrm{FP} + \mathrm{FN} \tag{6.5b}$$

$$S = \frac{\mathrm{TP}+\mathrm{FN}}{N} \tag{6.5c}$$

$$P = \frac{\mathrm{TP}+\mathrm{FP}}{N} \tag{6.5d}$$

where F_β is the F-beta score parametrized by β, CBA is the class balance accuracy, MCC is the Matthews correlation coefficient, and TP, TN, FP, and FN are the true positive, true negative, false positive, and false negative counts respectively. Increasing β in F_β decreases the weight of the FP and TN relative to the TP and FN. Unlike the overall classification accuracy, all these measures can provide less biased estimate of the predictive accuracy on tasks involving highly imbalanced class distribution. Aside from these three measures, we also report three important numbers in the confusion matrix when dealing with highly imbalanced binary classes: the TP, FN, and FP.

All the metrics reported are calculated on test patches of size 128×128 pixels. We recognize that evaluating a pixel-wise accuracy is too restrictive to predict a very rare phenomenon that greatly varies in size from one pixel to tens of thousands. Therefore, we adapted a less-constrained measure allowing

a tolerance of eight pixels. The measure is equivalent to applying 8×8 non-overlapping maximum pooling filters to both the predictions and reference before calculating the accuracy metrics. The size of the tolerance, equivalent to around 16 km^2 in ground area, is comparable to the sizes of recommended planned burned area units by several case studies reported in the Australian National Guidelines for Prescribed Burning Operations [4].

6.2.6 Measuring statistical importance of input variable

Another issue of interest is the relative statistical importance of each input feature in the prediction task. This could help future analysis and data collection efforts to choose which dataset to prioritize when considering a similar case study. For this we consider two measures: 1) the average gradient with respect to the input feature [143] for the three deep neural networks and 2) the LR coefficients.

Table 6.3 Selected network hyperparameter values

Model	M	L	K	γ
AllConvNet	32	21	32	100
SegNet	64	—	—	5
MLP	—	4	64	20

6.2.7 Network hyperparameters

We experimented with varying hyperparameters of the three deep neural networks including: the input patch size M for both AllConvNet and SegNet; the number of layers L and the number of filters in each convolutional layer K for both MLP and AllConvNet; and the class weighting value γ for all three networks. Values for each hyperparameter were chosen by optimizing the network accuracy on a validation set (taken from within the training period but is spatially disjoint from the training set). The parameters yielding the best validation accuracy for each network are shown in Table 6.3.

6.3 Results and Discussion

6.3.1 Burn prediction maps

We visually compare the resulting predicted maps from each of our four predictive models. Performance on contrasting days—e.g., during summer when a wildfire is recorded and winter when no fire is recorded—cannot be inferred just by observing Tables 6.4 and 6.5. We, therefore, visualize three sets of sample maps resulting from our four models' predictions. Firstly, a sampled date (date I) within the training period (December 1, 2006) having recorded a large wildfire (around 680,000 hectares); secondly, a sampled date (date II) within the test period (October 18, 2017) also with recorded, but relatively smaller than the previous date, wildfire (around 8,000 hectares); and lastly, a sampled date (date III) within the test period (June 27, 2017) without any recorded wildfire.

Figure 6.6 shows the reference and predicted maps for date I. Figure 6.7 shows the reference and predicted maps for date II. Figure 6.8 shows the reference and predicted maps for date III. Looking at each set of results from a model across the three dates, we can see that the amount of predicted wildfire-prone locations (red pixels) increases together with the area of wildfire observed for a date, except for SegNet and less noticeable for LR. AllConvNet and LR perfectly predicts the locations without wildfire for a given date as shown in Figure 6.8. This observation suggests that the AllConvNet and LR are relatively better at temporally distinguishing whether a day is more wildfire-prone compared to another day. Spatially comparing the two dates with recorded wildfire, the predictions of wildfire-prone locations appears to concentrate on comparably similar locations for each of the classifier, with observable size and pattern changes being more apparent in the results of AllConvNet and SegNet.

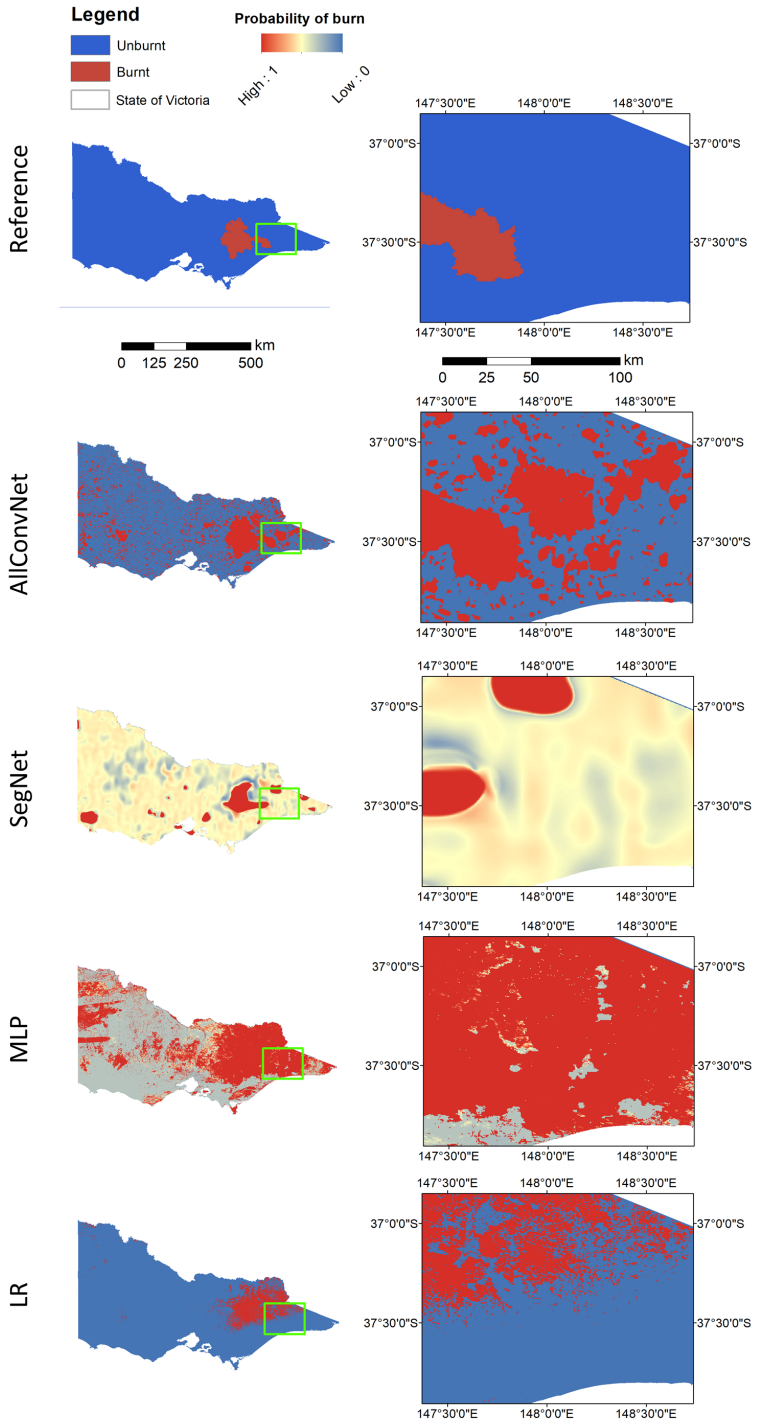


Figure 6.6 Visualization of reference (topmost row) and predicted maps for a date (December 1, 2006) when a wildfire was recorded. The maps to the left show the prediction for the whole extent of Victoria while to the right is an inset map of a zoomed location. The maps use WGS84 as their coordinate reference system.

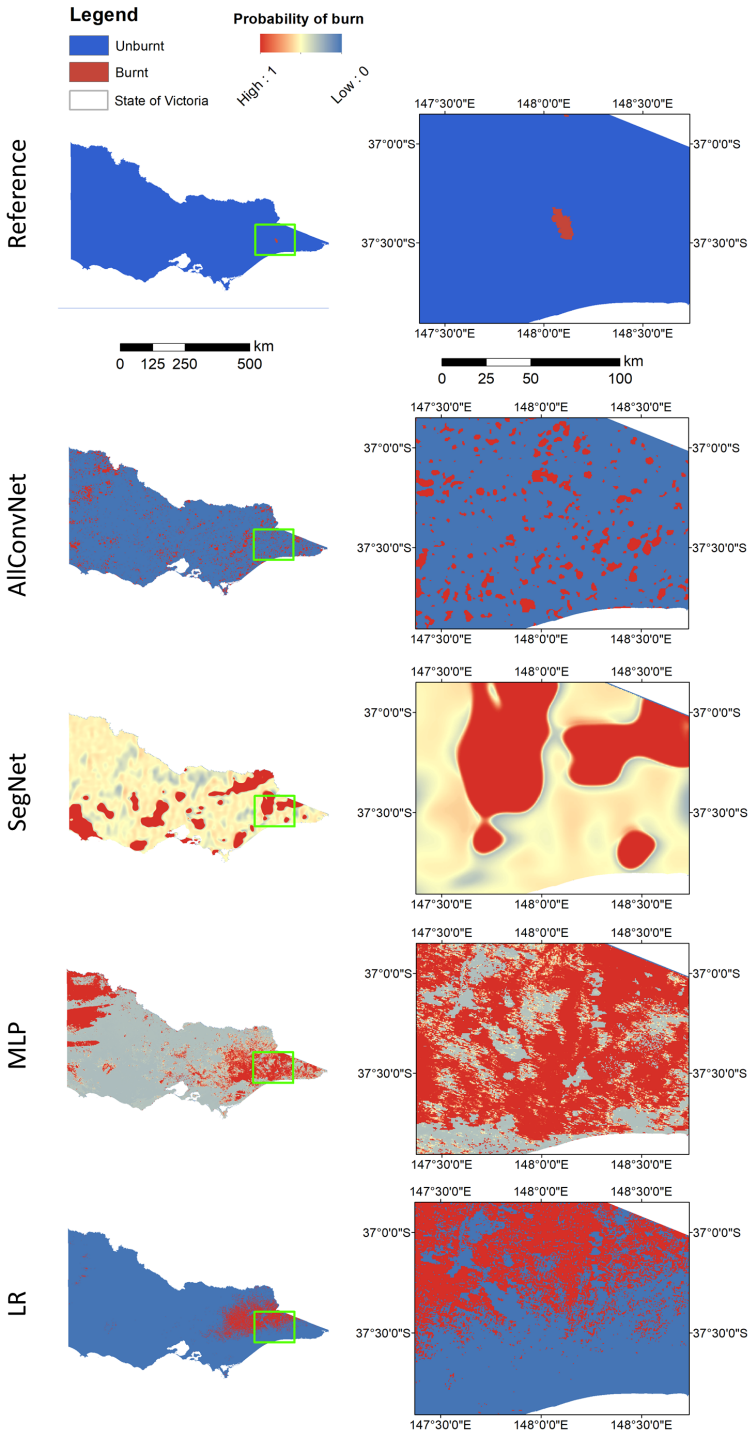


Figure 6.7 Visualization of reference (topmost row) and predicted maps for a date (October 18, 2017) when a wildfire was recorded. The maps to the left show the prediction for the whole extent of Victoria while to the right is an inset map of a zoomed location. The maps use WGS84 as their coordinate reference system.

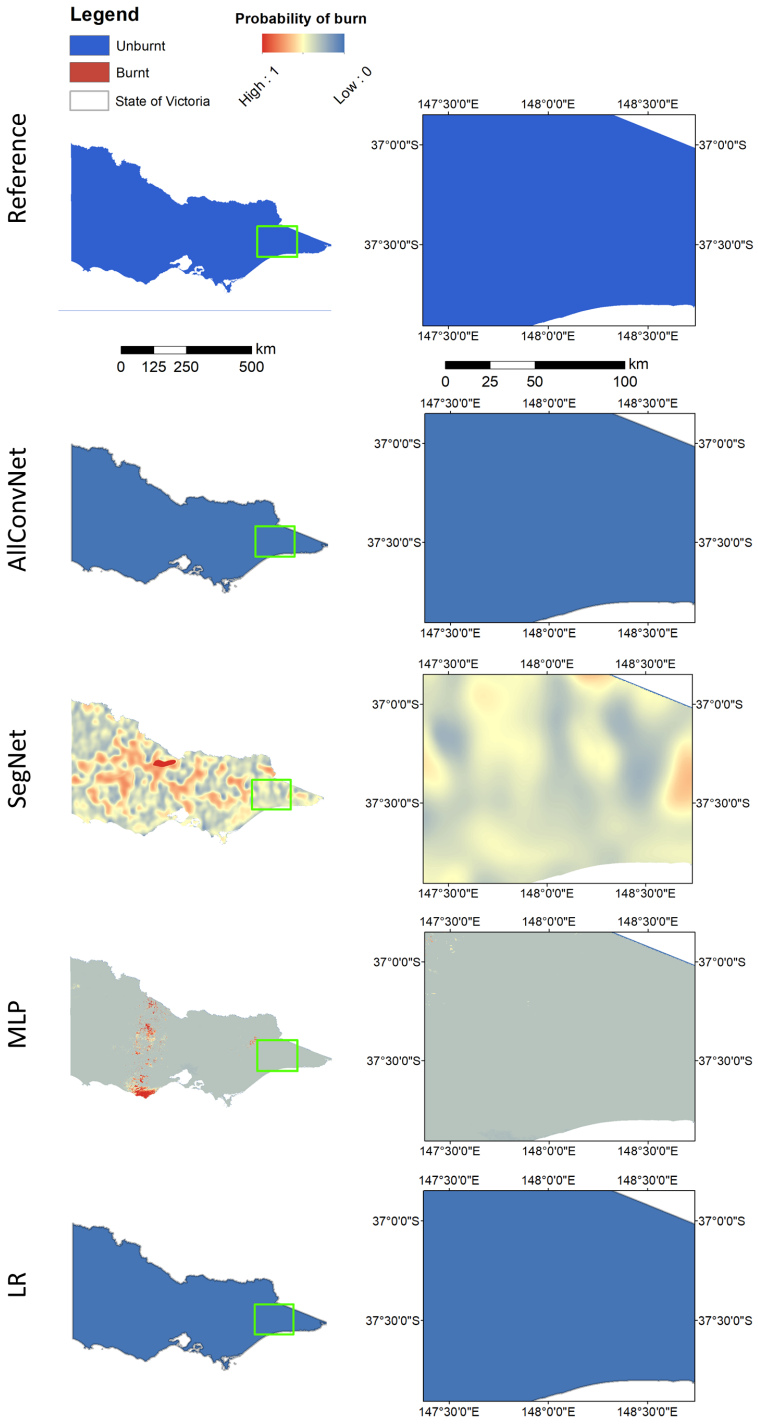


Figure 6.8 Visualization of reference (topmost row) and predicted maps for a date (June 27, 2017) when no wildfire was recorded. The maps to the left show the prediction for the whole extent of Victoria while to the right is an inset map of a zoomed location. The maps use WGS84 as their coordinate reference system.

We also notice a significant portion of the maps predicted for dates I and II appear to have a high number of FP's. Since the phenomenon can be generally observed regardless of the model, then a possible explanation of this can be the lack of discriminatory information encoded in our input features. We can decompose the probability of a wildfire burning into the probability of fuel ignition, given a causative agent triggers an ignition, and the probability of a causative agent starting an ignition. On the one hand, the information on the first probability is intuitively embedded in the features describing fuel characteristics such as topography, land cover, fuel moisture proxies, and the meteorological observation. On the other hand, only the lightning flash density rate and proximity measure (from roads and power lines) features generally embeds information on the second probability—both of which do not have daily temporal resolution. Observing the sample map predictions, all the models seem to capture the interactions within the first probability stronger, that is to say, the models are able to spatially distinguish the differences in fuel ignitability more than the differences in whether one location is more likely to experience an ignition than others. Adding features embedding information on the second probability, if possible, with higher temporal resolution, might help address this challenge.

Both the convolutional networks predict more connected fire-prone regions compared to the results of MLP and LR that shows noisier predictions—with the results from the LR and SegNet at the extreme opposing ends, SegNet producing the smoother and more regularized results. This can be explained by the fact that the two convolutional networks accepts as an input $M \times M$ patches, a practice that is rarely done in similar case studies, as opposed to single pixel vectors utilized as an input by LR and MLP. By accepting input patches instead of single pixel vectors, the trainable convolutional filters of the two networks allow the latter to learn spatial-contextual dependencies that may be present between the input features.

Another notable observation is the difference on how the models assign final class score, $p(burn|x = X)$. In the one instance, AllConvNet and LR assigning more extremely high scores to most, if not all, wildfire-prone regions. Whereas conservative (in between extremely high and low) scores are more observable in the results of SegNet and MLP. Similar results were observed by **(author?)** [33] where the authors compared MLP and LR for wildfire ignition prediction—with the MLP producing more intermediate probability values. Extreme scores can be attributed to possible model overtraining on a dataset with limited positive samples such as recorded wildfire events. The cross-entropy loss function in Equation 6.2 continues to be minimized by assigning higher scores to already correctedly classified training samples instead of correctly classifying samples

which are still misclassified—hence, promoting high scores assigned to wildfire-prone locations during model testing. Broadly speaking, the maps from the two convolutional networks appear to be smoother and more regularized, with the SegNet having smoother and more regularized areas and providing more information by discriminating intermediate, less wildfire-prone areas.

Table 6.4 Comparison of estimated predictive model accuracy based on sample counts

Model	TP	FN	FP
AllConvNet	**538**	**390**	66881
SegNet	112	816	**18873**
MLP	412	516	69997
LR	167	761	24034

6.3.2 Predictive accuracy

Tables 6.4 and 6.5 show the results of our model compared to the three baseline models. We first report metrics based on sample counts in Table 6.4. Secondly, we report metrics based on class-averaged rates in Table 6.5. All the metrics are evaluated using results obtained by applying a threshold of 0.5 to the final class score, $p(burn|x = X)$, given by each predictive model.

AllConvNet has the highest TP and lowest FN counts but ranks 2nd lowest in terms of FP counts. On the contrary, SegNet produces the lowest FP counts but has the worst TP counts—only correctly classifying 12% of the positive samples as compared to the 58% of the AllConvNet. AllConvNet correctly classified 14% more positive samples than the MLP while still having 5% less FP samples. LR correctly classified 6% more positive samples than the SegNet but have 27% more false positives.

An ideal predictive model should naturally have high TP count and both low FP and FN counts. None of the four models outperforms the rest in all three counts with the MLP and SegNet being on the extreme ends—the MLP overpredicting and the LR underpredicting positives. Choosing the best among the four would be a matter of how much relative importance we assign to each of the three counts. We argue that putting more weight on both the TP and FN than the FP sounds more reasonable in the context of hazard-related studies such as wildfire prediction. In hazard-related studies where we predict a rare phenomenon, FP's can either be actual miscategorization or possibly genuine hazard-prone locations that are yet to develop the hazardous phenomenon [10]. In line with the argument of assigning less relative weight

Table 6.5 Comparison of estimated predictive model accuracy based on averaged rates

Model	F_1	F_5	F_{20}	F_{100}	CBA	MCC
AllConvNet	37.64%	**56.20%**	**74.39%**	**78.57%**	**58.23%**	0.025
SegNet	**50.56%**	53.45%	55.78%	56.02%	56.03%	**0.027**
MLP	36.63%	54.03%	64.33%	71.75%	50.48%	0.002
LR	46.52%	53.98%	58.17%	58.70%	51.54%	0.007

on FP (recall-biased premise), we report additional F_β measures in Table 6.5 aside from the conventional F_1 measure. Moreover, we report the CBA and MCC which are specifically developed for assessing problems with imbalanced class distribution.

SegNet achieves the highest F_1 and MCC while AllConvNet outperforms the rest of the models in terms of the four other measures. All F_β scores consistently increase for all the models as the β increases. The β parameter intuitively provides a way, in the model assessment step, to control how much importance we want to assign on the model's capability to predict as much as wildfire burn event as it can while being more tolerant on false alarms. However, the choice of such weighting will generally fall on the fourth stage of a wildfire risk assessment framework, risk characterization, that is better handled by land and fire management agencies in line with their management objectives.

In these experiments, we evaluated the models varying β to be equal to the optimal γ values chosen in the hyperparameter selection steps (see Table 6.3), since $\gamma > 1$ serves a similar purpose as β but is employed in the training phase of the models. Intuitively, assigning higher γ value would influence the model to overpredict, increasing FP. However, results presented in Table 6.4 demonstrates that AllConvNet, despite having higher γ than MLP, is less prone to overprediction. While scoring highest in terms of all F_β except F_1 and the parameter-free CBA, AllConvNet also comes out second in terms of the other parameter-free measure MCC. This shows that improvement in the accuracy metrics of the results from AllConvNet is independent from our recall-biased premise.

6.3.3 Feature statistical importance

Artificial neural networks are often criticized for being less interpretable than other models like LR, such that, it is difficult to determine the internal statistical importance of each input variable being considered in the modelling task. This

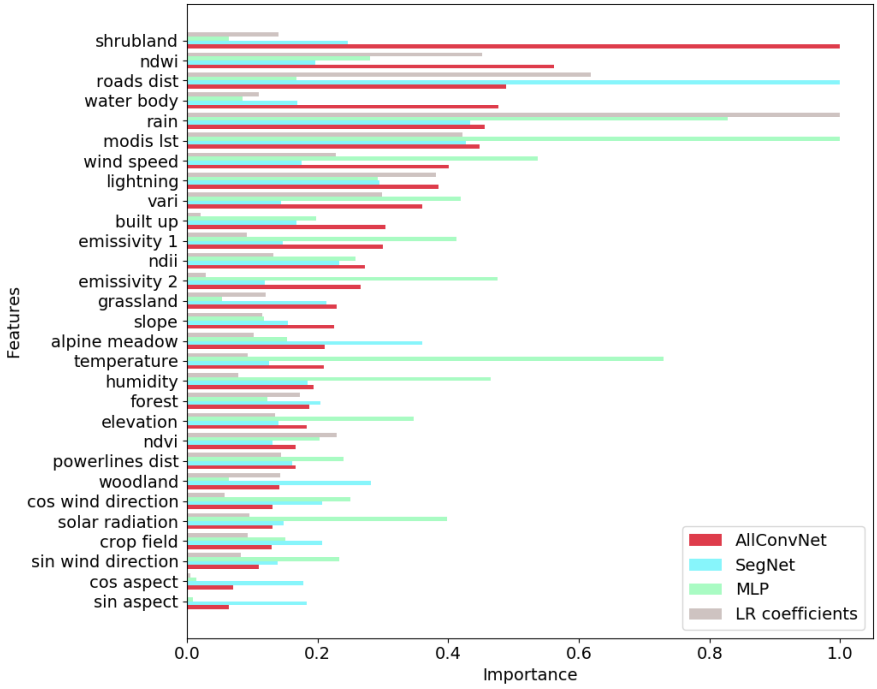


Figure 6.9 Feature statistical importance measures considering gradient with respect to the input feature for AllConvNet, SegNet, and MLP and the normalized magnitude of the LR coefficients.

apparent low model transparency can be negligible in the context of other predictive tasks, e.g. in land cover classification where there may be less significance in knowing how the digital values assigned to pixels in a satellite image get transformed into land cover categories. In the context of wildfire prediction, understanding the relative statistical importance of each input variable as to how they affect the predictability of a wildfire event can be valuable as most fire danger ratings, such as the McArthur Forest Fire Danger Index and the Canadian Forest Fire Weather Index, are similarly based on relative importance weighting of wildfire-related variables. Figure 6.9 shows the feature statistical importance measures considering two techniques: 1) gradient with respect to the input feature for AllConvNet, SegNet, and MLP and 2) normalized magnitude of the LR coefficients.

Table 6.6 Correlation of the feature statistical importance measures

Model A	Model B	Pearson's R	p-value
AllConvNet	SegNet	0.32	0.0858
AllConvNet	MLP	0.15	0.4288
AllConvNet	LR coefficients	0.45	0.0146
SegNet	MLP	0.08	0.6947
SegNet	LR coefficients	0.62	0.0002
MLP	LR coefficients	0.47	0.0097

Total precipitation, lightning flash density, and land surface temperature occur to be consistently highly weighted by all models while terrain aspect components, wind direction components, certain land cover classes (such as crop field and woodland), and distance from power lines are ranked on the lower end. In a study employing MLP and LR to predict fire weather index, [143] compared a subset of the weather variables that we considered in our own study and found that their model also ranked the amount of precipitation as the highest among four variables they compared followed by temperature, wind speed, and finally relative humidity—a result comparable to what we can observe from the ranking provided by AllConvNet.

The slope and elevation and presence of certain land cover classes (forest, alpine meadow, and grassland) were given moderate importance by AllConvNet together with most of the bands and indices derived from MODIS except NDWI. Slope and elevation ranked consistently higher than the terrain aspect components. MODIS derived land surface temperature has higher statistical importance measure than the two emmissivity bands. Notably, NDWI and VARI generally ranks higher than NDVI and NDII even though [18], regressing these four indices against live fuel moisture content, found NDII and NDWI to have the highest and lowest coefficient of determination respectively.

The degree of agreement between the resulting set of feature statistical importance scores cannot be quantitatively observed in Figure 6.9. Thus, we present Table 6.6 showing the correlation coefficients and corresponding p-values of the feature statistical importance scores shown in Figure 6.9. Statistical importance measures from logistic regression appear to have highest correlation values with the other models, MLP generally having the lowest ones. SegNet has the lowest and highest correlation values with LR and MLP respectively. This could be due to the increased model complexity of MLP but still having the same input dimensionality as LR, possibly making the former more prone

to fixating on features the other models found less statistically important.

6.4 Conclusion

This study presents a deep fully convolutional neural network for predicting daily maps of the probability of a wildfire burn over the next 7 days in Victoria, Australia for the period of 2006 - 2017. The proposed network architecture, AllConvNet, outperforms the other three baseline methods namely: SegNet, multilayer perceptron, and logistic regression in four of the six quantitative metrics (four recall-biased F-beta scores, class balance accuracy score, and Matthews correlation coefficient) and ranks second in one of the other two parameter-free metric. Both the two convolutional networks, AllConvNet and SegNet, also provide smoother and more regularized predicted maps, with SegNet providing better visual information on discriminating less wildfire-prone locations. Input feature statistical importance was measured for the three networks and compared against logistic regression coefficients. Total precipitation, lightning flash density, and land surface temperature occurs to be consistently highly weighted by all models while terrain aspect components, wind direction components, certain land cover classes (such as crop field and woodland), and distance from power lines are ranked on the lower end.

The deep fully convolutional networks designed in this study demonstrate better predictive accuracy and map quality than the baseline methods commonly explored in previous studies. Exploratory feature statistical importance measures presented in this work also provide a way to better understand these rather less transparent models. Future studies on the application of deep learning for wildfire prediction can look into improving the features encoding information on the probability that a location would experience an ignition. In the context of deep learning, future works on incorporating temporal information, accepting as inputs and producing as outputs sequences instead of daily snapshots of features and predictions, and data fusion within the model design would be relevant. Resulting maps from our models can be reclassified into wildfire burn exposure indices and be intersected with information on assets-at-risk to produce alternative wildfire risk index maps.

Synthesis

7

7.1 Research findings and conclusions

This research focuses on deep learning based methods for two relevant prediction problems in remote sensing including: urban land cover and land use classification, and mapping the probability of wildfire burn events. Several components of standard prediction processes in all the use cases—such as automated feature extraction, image fusion, post-classification regularization, quantification of input feature importance—were all improved and optimized by embedding them within an end-to-end framework. Below I summarize the main findings of the research objectives presented in Section 1.4. Section 7.2 reflects on the implications of this work and discusses future research directions.

Research objective 1: To develop a deep learning based method performing an end-to-end image fusion and classification of a multiresolution VHR satellite image in the context of urban land cover classification. Results showed that there can be an improvement fo 3–10% in the classification accuracy of predicted urban land cover maps when opting to perform image fusion and feature extraction within a multiresolution convolutional network, FuseNet, as opposed to performing a separate image fusion step utilizing a fixed or learnable filters. Results on varying the architecture of the proposed network also indicated that performing the fusion on the resolution of the multispectral image yields a higher classification accuracy than performing the fusion on the higher resolution of the panchromatic image. A sensitivity analysis of the network hyperparameters showed that it is more important to choose the proper upsampling operation, patch size, and corresponding bottleneck feature dimension of the network than varying the number of hidden layers. The utility of a single-stage classification pipeline incorporating image fusion and feature extraction combined within a convolutional network trained in an end-to-end manner improves the land

cover classification process.

Research objective 2: To develop a deep learning based method to model contextual label-to-label dependencies and effectively regularize classification maps in the context of urban land cover classification. Experimental results from the proposed recurrent multiresolution convolutional network, ReuseNet, demonstrates the improvement on classification accuracy of urban land cover from very high resolution imageries by incorporating contextual label dependency as similarly done by other post-classification regularization methods like conditional random fields. Land cover maps classified by ReuseNet also show a smoother, less noisy classification than those obtained by classifiers that do not utilize any contextual label information. Aside from streamlining the classification process, by integrating several conventionally independent steps in the prediction pipeline (such as image fusion, feature extraction, classification, and map regularization), ReuseNet also shows 0.5–2.0% increase on classification accuracy over the baseline methods. The initialization of ReuseNet's parameters and initial score maps can also greatly affect the classification accuracy of the network, changing it by as much as 12%. Inclusion and leverage of contextual label information is separate from the design of the base network architecture, and can therefore be incorporated to other state-of-the-art network architectures.

Research objective 3: To develop a deep learning based method to classify urban land use from VHR satellite images. Results in this chapter show at least 30% increase in classification accuracy of the urban land cover classes when using the two multitask networks (PMN and SMN) over the other standard classification methods. The two multitask networks also improves the predictions on the additionally embedded land cover classification task. Performing land use classification simultaneously with land cover classification within an end-to-end deep multitask network improves the classification performance on both the main urban land use classification and the complimentary land cover classification tasks.

Research objective 4: To develop a deep learning based method predicting daily maps of the probability of a wildfire burn. The proposed network architecture, AllConvNet, outperforms the other classifiers in four of the six quantitative metrics (four recall-biased F-beta scores, class balance accuracy score, and Matthews correlation coefficient) considered in this study. Experiments on quantifying input feature statistical importance show that total precipitation, lightning flash density, and land surface temperature are consistently highly weighted by all the models while terrain aspect components, wind direction components, certain land cover classes (such as crop field and woodland), and distance from power lines are ranked on the lower end. Using a deep

convolutional network improves the accuracy of daily probability of wildfire burn maps over standard logistic regression and multilayer perceptron. Quantification of input feature statistical importance allows the interpretation of models considered by many end-users as "black boxes" like deep convolutional networks.

In sum, this research contributed towards the improvement of state-of-the-art methods for automated feature extraction, image fusion, classification, and post-classification regularization in the context of urban land cover mapping; land use classification; and wildfire burn prediction. Five different end-to-end models: FuseNet, ReuseNet, PMN and SMN, and AllConvNet, were developed to address three different use cases. Results from these models show quantitative and qualitative improvements over results from state-of-the-art methods.

7.2 Reflections and recommendations

The research was formulated with the aim of developing end-to-end predictive models that can contribute towards mapping technologies leveraging remote sensing data. I address this by formulating techniques within the deep learning paradigm. Models developed from this research was able to blend a diverse group of dataset—varying in spatial, spectral, temporal resolution and modality—to generate predictions for three remote sensing applications: urban land cover classification, urban land use classification, and wildfire prediction. Deep learning techniques made it possible to construct these predictive models in an end-to-end manner, generating predictions directly from the input data by integrating conventionally separate processing steps.

Convolutional networks, found to be effective in dealing with many computer vision problems, were adapted and extended to address several issues intrinsic to remote sensing problems. Chapter 3 tackles the image fusion problem inherent with classifying multiresolution VHR imagery. Chapter 4 presents a novel way to incorporate contextual label information to improve classification results in a similar manner that a separate post-classification regularization technique would do. Chapter 5 leverages similar remote sensing tasks, i.e. land cover classification as a complimentary task to land use classification, to boost the classification accuracy on highly abstracted classes such as urban land use. And chapter 6 introduces a method based on deep convolutional networks to map daily probability of wildfire burn, and consequently quantify input feature statistical importance—presenting a way to unravel the transparency of what many end-users consider as obscure deep learning models.

Building end-to-end models streamlines and objectifies the prediction process by integrating separate independent steps within one model and directly coupling these steps to the objective function formulated for the specific prediction problem. However, such improvements come with a few drawbacks such as: i) increased training time, ii) obfuscation of the "physical interpretation" of the model, iii) and the need to provide a relatively larger number of labeled training samples. Both the first two drawbacks are observed in all the experiments from Chapters 3 to 6. Training can be done separately from operationalization, thus, this issue can either be neglected or directly addressed depending on the scope and goals of a project. The second drawback depends on the nature of the problem, whether the physical interpretation of intermediate features have impact on the objectives of the problem. For urban land cover and land use classification, I would argue that there is little value to such physically interpreting digital numbers extracted from an optical image to come up with appropriate urban land cover and land use labels. However, for wildfire risk related application such as that presented in Chapter 6, the possible physical interpretation and connection of intermediate learned features to variables intuitively related to wildfire risk but is not captured by the input data can be beneficial. The need for a larger number of labeled samples was also encountered in Chapters 3 to 6. This issue can only be addressed if there is a way to collect additional samples, i.e. in urban land cover and land use classification, by investing more resources on data collection efforts, otherwise can be ignored when there is no way to collect additional samples, i.e. wildfire mapping application.

Taking into account these drawbacks, other use cases that can benefit from adapting the methodologies developed in this research are those that are more concerned with increasing the level of automation and accuracy of a prediction task rather than the explanation and interpretation of how the predictions are obtained from the input data. On the one hand, tasks like producing accurate masks for cloud and cloud shadow and mapping locations with dense water vegetation as parts of a project monitoring complex waterways could benefit more on the aspect of automation and accuracy rather than model interpretability. On the other hand, spatiotemporally mapping agricultural activities for the purpose of regulating statutory subsidies for farmers would require more transparency and interpretability from the model. The resource related constraints on training time and number of labeled samples also limits the spatial and temporal extent with respect to spatial and temporal resolution of the use cases adapting these methods. Applications which require very fine resolution outputs, e.g. weekly monitoring narrow waterways, will likely be bounded on the spatial extent of the area of interest (possibly a small-sized

city) for computational and practical reasons.

The second example opens up another kind of problems when using methods based on deep learning that is: the ethical implications of models that are not fully physically explainable. In the agricultural activity monitoring example, decisions derived from the models can affect the livelihood of an individual and therefore potential petitions challenging these decisions can arise. Hence, a model incapable of fully explaining how the predictions were produced can become problematic for the end users of these predictions. This also questions the accountability for decisions made from possibly erroneous model predictions. Therefore, for such situational applications, combining guided manual checks, e.g. field visits on flagged nonconforming parcels, and automated methods will be the way to go.

The studies presented in Chapters 3 to 5 illustrates a rather more familiar remote sensing application utilizing multispectral very high resolution images to provide geospatial information on the urban environment. Spectral bands (RGB and NIR) and target classes (land cover and land use) are widely employed in other similar research problems. This enables straightforward adaptation of the methods we have developed in these chapters. On the other hand, Chapter 6 presented a relatively niche application were there is no standard treatment of inputs and formalization of outputs. Comparison between different but related studies are therefore complicated and the research contributes more towards the analysis of the input variables and the assessment of the quality of output predictions than the method development itself. This second to the last chapter balances out the previously presented chapters heavily focused on method development and assessment.

The methods developed in this study can be extended and adapted to similar studies that, for example, are focusing on study areas involving a different kind of environment such as agricultural, glacial, or transition areas like coastal shores. The methods in Chapters 3 to 5 can also be embedded within an operational workflow mapping change signals for automatic updating of topographical or other geospatial databases for the urban environment. Similarly, the outputs of Chapter 6 can either directly serve as a proxy measure for wildfire risk or may serve as input, together with information about assets-at-risk and their corresponding vulnerabilities, to probabilistic methods from the actuarial sciences to quantify wildfire risk. Furthermore, these outputs can also guide problems on wildfire response and fuel management, e.g. probability of wildfire burn as an input to prescribed burning optimization. On the methods development side, the network presented in Chapter 6 can be further extended accommodate additional input features and explicitly incorporate temporal relationships of the latter within the network architecture. Further exploration

of a Bayesian treatment of the deep networks developed in this study can provide a way to explicitly quantify the uncertainties of the output predictions of our models.

Bibliography

[1] A. A. Ager, M. A. Day, C. W. McHugh, K. Short, J. Gilbertson-Day, M. A. Finney, and D. E. Calkin. Wildfire exposure and fuel management on western US national forests. *Journal of Environmental Management*, 145:54–70, 2014.

[2] M. Alioscha-Perez and H. Sahli. Efficient learning of spatial patterns with multi-scale conditional random fields for region-based classification. *Remote Sensing*, 6(8):6727–6764, 2014.

[3] D. Atkinson, M. Chladil, V. Janssen, and A. Lucieer. Implementation of quantitative bushfire risk analysis in a GIS environment. *International Journal of Wildland Fire*, 19(5):649–658, 2010.

[4] Australasian Fire Emergency Service Authorities Council and Forest Fire Management Group. *National Guidelines for Prescribed Burning Operations*. Australasian Fire Emergency Service Authorities Council, Level 1, 340 Albert Street East Melbourne Victoria 3002, 2014.

[5] A. Badia, P. Serra, and S. Modugno. Identifying dynamics of fire ignition probabilities in two representative Mediterranean wildland-urban interface areas. *Applied Geography*, 31(3):930–940, 2011.

[6] V. Badrinarayanan, A. Kendall, and R. Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2017.

[7] G. H. Bakir, T. Hofmann, B. Schölkopf, A. J. Smola, B. Taskar, and S. V. N. Vishwanathan. *Predicting Structured Data (Neural Information Processing)*. The MIT Press, 2007.

[8] G. Ball and D. Hall. Isodata: A novel method of data analysis and pattern classification. Technical report, Stanford Research Institute, Menlo Park, 1965.

[9] A. Bar Massada, V. C. Radeloff, S. I. Stewart, and T. J. Hawbaker. Wildfire risk in the wildland-urban interface: A simulation study in northwestern Wisconsin. *Forest Ecology and Management*, 258(9):1990–1999, 2009.

[10] S. Beguería. Validation and Evaluation of Predictive Models in Hazard Assessment and Risk Management. *Natural Hazards*, 37(3):315–329, March 2006.

[11] Y. Bengio. Learning deep architectures for AI. *Foundations and trends in Machine Learning*, 2(1):1–127, 2009.

[12] J. R. Bergado. A deep feature learning approach to urban scene classification. Master's thesis, The University of Twente, 2016.

[13] J. R. Bergado, C. Persello, and A. Stein. Recurrent multiresolution convolutional networks for vhr image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 56(11):6361–6374, Nov 2018.

[14] U. Bhandary and B. Muller. Land use planning and wildfire risk mitigation: an analysis of wildfire-burned subdivisions using high-resolution remote sensing imagery and GIS data. *Journal of Environmental Planning and Management*, 52(7):939–955, 2009.

[15] C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006.

[16] L. Bottou. Stochastic gradient descent tricks. In *Neural Networks: Tricks of the Trade*, pages 421–436. Springer Berlin Heidelberg, 2012.

[17] Y.-L. Boureau, J. Ponce, and Y. Lecun. A theoretical analysis of feature pooling in visual recognition. In *27th International Conference on Machine Learning*, 2010.

[18] G. Caccamo, L. A. Chisholm, R. A. Bradstock, M. L. Puotinen, and B. G. Pippen. Monitoring live fuel moisture content of heathland, shrubland and sclerophyll forest in south-eastern Australia using MODIS data. *International Journal of Wildland Fire*, 21(3):257, 2012.

[19] J. Cancelo-González, C. Cachaldora, F. Díaz-Fierros, and B. Prieto. Colourimetric variations in burnt granitic forest soils in relation to fire severity. *Ecological Indicators*, 46:92–100, 2014.

[20] F. X. Catry, F. C. Rego, F. L. Bação, and F. Moreira. Modeling and mapping wildfire ignition risk in Portugal. *International Journal of Wildland Fire*, 18(8):921, 2009.

[21] D. Cecil, D. Buechler, and R. Blakeslee. Lis/otd gridded lightning climatology data collection. `https://ghrc.nsstc.nasa.gov/`, 2014. Accessed: 2018-10-22.

[22] G. Chen, M. R. Metz, D. M. Rizzo, W. W. Dillon, and R. K. Meentemeyer. Object-based assessment of burn severity in diseased forests using high-spatial and high-spectral resolution master airborne imagery. *ISPRS Journal of Photogrammetry and Remote Sensing*, 102:38 – 47, 2015.

[23] L. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *CoRR*, abs/1606.00915, 2016.

[24] X. Chen, R. Tateishi, and C. Wang. Development of a 1-km landcover dataset of China using AVHRR data. *ISPRS Journal of Photogrammetry and Remote Sensing*, 54(5–6):305–316, 1999.

[25] Z. Chen, Y. Zhang, B. Guindon, T. Esch, A. Roth, and J. Shang. Urban land use mapping using high resolution SAR data based on density analysis and contextual information. *Canadian Journal of Remote Sensing*, 38(6):738–749, 2012.

[26] M. Chini, F. Pacifici, and W. J. Emery. Morphological operators applied to X-band SAR for urban land use classification. In *2009 IEEE International Geoscience and Remote Sensing Symposium*, volume 4, pages IV–506–IV–509, 2009.

[27] E. H. Chowdhury and Q. K. Hassan. Operational perspective of remote sensing-based forest fire danger forecasting systems. *ISPRS Journal of Photogrammetry and Remote Sensing*, 104:224 – 236, 2015.

[28] E. Chuvieco, I. Aguado, M. Yebra, H. Nieto, J. Salas, M. Martín, L. Vilar, J. Martínez, S. Martín, P. Ibarra, J. de la Riva, J. Baeza, F. Rodríguez, J. Molina, M. Herrera, and R. Zamora. Development of a framework for fire risk assessment using remote sensing and geographic information system technologies. *Ecological Modelling*, 221(1):46–58, 2010.

[29] D. Clevert, T. Unterthiner, and S. Hochreiter. Fast and accurate deep network learning by exponential linear units (ELUs). In *International Conference on Learning Representations*, 2016.

[30] R. Congalton and K. Green. *Assessing the Accuracy of Remotely Sensed Data: Principles and Practices*. Mapping Science Series. CRC Press/Taylor & Francis, 2009.

[31] M. Cramer. The DGPF-test on digital airborne camera evaluation — overview and test design. *Photogrammetrie - Fernerkundung - Geoinformation*, 2:73–82, 2010.

[32] P. P. de Bem, O. A. de Carvalho Júnior, E. A. T. Matricardi, R. F. Guimarães, and R. A. T. Gomes. Predicting wildfire vulnerability using logistic regression and artificial neural networks: a case study in Brazil. *International Journal of Wildland Fire*, 28:35–45, 2018.

[33] M. P. de Vasconcelos, S. Silva, M. Tome, M. Alvim, and J. C. Pereira. Spatial prediction of fire ignition probabilities: comparing logistic regression and neural networks. *Photogrammetric engineering and remote sensing*, 67(1):73–81, 2001.

[34] L. Deng and D. Yu. Deep learning: Methods and applications. Technical report, Microsoft, May 2014.

[35] A. M. Dewan and Y. Yamaguchi. Land use and land cover change in Greater Dhaka, Bangladesh: Using remote sensing to promote sustainable urbanization. *Applied Geography*, 29(3):390–401, 2009.

[36] K.-L. Du and M. N. S. Swamy. *Neural Networks and Statistical Learning*. Springer-Verlag, London, 2014.

[37] R. Dutta, A. Das, and J. Aryal. Big data integration shows Australian bush-fire frequency is increasing significantly. *Royal Society Open Science*, 3(2):150241, Feb. 2016.

[38] A. Fairbrother and J. G. Turnley. Predicting risks of uncharacteristic wildfires: Application of the risk assessment process. *Forest Ecology and Management*, 211(1–2):28–35, 2005.

[39] S. Fang, L. D. Xu, Y. Zhu, J. Ahati, H. Pei, J. Yan, and Z. Liu. An integrated system for regional environmental monitoring and management based on internet of things. *IEEE Transactions on Industrial Informatics*, 10(2):1596–1605, 2014.

[40] M. Fauvel, Y. Tarabalka, J. A. Benediktsson, J. Chanussot, and J. C. Tilton. Advances in spectral-spatial classification of hyperspectral images. *Proceedings of the IEEE*, 101(3):652–675, March 2013.

[41] R. Fernandes and S. G. Leblanc. Parametric (modified least squares) and non-parametric (Theil-Sen) linear regressions for predicting biophysical parameters in the presence of measurement errors. *Remote Sensing of Environment*, 95(3):303–316, 2005.

[42] M. A. Finney. The challenge of quantitative risk analysis for wildland fire. *Forest Ecology and Management*, 211(1–2):97–108, 2005.

[43] G. M. Foody. Status of land cover classification accuracy assessment. *Remote Sensing of Environment*, 80(1):185–201, 2002.

[44] K. Fukushima. Neocognitron: A hierarchical neural network capable of visual pattern recognition. *Neural Networks*, 1(2):119–130, 1988.

[45] K. Fukushima and S. Miyake. Neocognitron: A new algorithm for pattern recognition tolerant of deformations and shifts in position. *Pattern Recognition*, 15(6):455–469, 1982.

[46] A. D. Giorgi, G. Moser, and S. B. Serpico. Contextual remote-sensing image classification through support vector machines, markov random fields and graph cuts. In *2014 IEEE Geoscience and Remote Sensing Symposium*, pages 3722–3725, July 2014.

[47] X. Glorot and Y. Bengio. Understanding the difficulty of training deep feedforward neural networks. In *Aistats*, volume 9, pages 249–256, 2010.

[48] X. Glorot, A. Bordes, and Y. Bengio. Deep sparse rectifier neural networks. In *Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics, AISTATS 2011, Fort Lauderdale, USA, April 11-13, 2011*, pages 315–323, 2011.

[49] I. Goodfellow. *Deep learning of representations and its application to computer vision*. PhD thesis, University of Montreal, Montreal, Canada, 4 2014.

[50] I. Goodfellow, Y. Bengio, and A. Courville. *Deep Learning*. MIT Press, 2016. `http://www.deeplearningbook.org`.

[51] I. J. Goodfellow, D. Warde-Farley, M. Mirza, A. C. Courville, and Y. Bengio. Maxout networks. In *Proceedings of the 30th International Conference on Machine Learning, ICML 2013, Atlanta, GA, USA, 16-21 June 2013*, pages 1319–1327, 2013.

[52] D. Graupe. Deep learning neural networks: Design and case studies, 2016.

[53] M. E. Gray, L. J. Zachmann, and B. G. Dickson. A weekly, continually updated dataset of the probability of large wildfires across western US forests and woodlands. *Earth System Science Data*, 10(3):1715–1727, Sept. 2018.

[54] Ç. Gülçehre, K. Cho, R. Pascanu, and Y. Bengio. Learned-norm pooling for deep feedforward and recurrent neural networks. In *Proceedings of the European Conference on Machine Learning and Knowledge Discovery in Databases - Volume 8724*, ECML PKDD 2014, pages 530–546, 2014.

[55] Z. Guo, W. Fang, J. Tan, and X. Shi. A time-dependent stochastic grassland fire ignition probability model for Hulun Buir Grassland of China. *Chinese Geographical Science*, 23(4):445–459, Aug. 2013.

[56] S. Hantson, M. Padilla, D. Corti, and E. Chuvieco. Strengths and weaknesses of modis hotspots to characterize global fire occurrence. *Remote Sensing of Environment*, 131:152–159, 2013.

[57] R. Haralick, K. Shanmugan, and I. Dinstein. Textural features for image classification, 1973.

[58] T. Hastie, R. Tibshirani, and J. Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition.* Springer Series in Statistics. Springer New York, 2009.

[59] K. He, X. Zhang, S. Ren, and J. Sun. Delving deep into rectifiers: Surpassing human-level performance on Imagenet classification. In *IEEE International Conference on Computer Vision (ICCV) 2015*, 2015.

[60] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 37(9):1904–1916, 2015.

[61] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Computer Vision and Pattern Recognition (CVPR), 2016 IEEE Conference on*, 2016.

[62] T. K. Hincks, B. D. Malamud, R. S. J. Sparks, M. J. Wooster, and T. J. Lynham. Risk assessment and management of wildfires. In J. Rougier, S. Sparks, and L. J. Hill, editors, *Risk and Uncertainty Assessment for Natural Hazards*, pages 398–444. Cambridge University Press, 2013. Cambridge Books Online.

[63] G. E. Hinton, N. Srivastava, A. Krizhevsky, I. Sutskever, and R. R. Salakhutdinov. Improving neural networks by preventing co-adaptation of feature detectors. *arXiv preprint arXiv:1207.0580*, 2012.

[64] S. Hu and L. Wang. Automated urban land-use classification with remote sensing. *International Journal of Remote Sensing*, 34(3):790–803, 2013.

[65] B. Huang, B. Zhao, and Y. Song. Urban land-use mapping using a deep convolutional neural network with high spatial resolution multispectral remote sensing imagery. *Remote Sensing of Environment*, 214:73–86, 2018.

[66] D. H. Hubel and T. N. Wiesel. Receptive fields, binocular interaction, and functional architecture in the cat's visual cortex. *Journal of Physiology (London)*, 160:106–154, 1962.

[67] Hutchinson, S. M. F., J. A. J. L., Stein, H. Anderson, and P. Tickle. Geodata 9 second dem and d8: Digital elevation model version 3 and flow direction grid 2008. `https://ecat.ga.gov.au/geonetwork/srv/eng/catalog.search?node=srv#/metadata/66006`, 2008. Accessed: 2018-09-26.

[68] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. *arXiv preprint arXiv:1502.03167*, 2015.

[69] A. Jaafari, D. M. Gholami, and E. K. Zenner. A Bayesian modeling of wildfire probability in the Zagros Mountains, Iran. *Ecological Informatics*, 39:32–44, May 2017.

[70] G. James, D. Witten, T. Hastie, and R. Tibshirani. *An Introduction to Statistical Learning: With Applications in R*. Springer Publishing Company, Incorporated, 2014.

[71] M. I. Jordan. Chapter 25 - serial order: A parallel distributed processing approach. In J. W. Donahoe and V. P. Dorsel, editors, *Neural-Network Models of Cognition Biobehavioral Foundations*, volume 121 of *Advances in Psychology*, pages 471–495. North-Holland, 1997.

[72] S. Jurdao and E. Chuvieco. Modelling Fire Ignition Probability from Satellite Estimates of Live Fuel Moisture Content. *Fire Ecology*, 7(1):77–97, Apr. 2012.

[73] I. Kamwa, S. R. Samantaray, and G. Joos. On the accuracy versus transparency trade-off of data-mining models for fast-response PMU-based catastrophe predictors. *IEEE Transactions on Smart Grid*, 3(1):152–161, March 2012.

[74] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, abs/1412.6980, 2014.

[75] P. Krähenbühl and V. Koltun. Efficient inference in fully connected crfs with gaussian edge potentials. In J. Shawe-Taylor, R. S. Zemel, P. L. Bartlett, F. Pereira, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 24*, pages 109–117. Curran Associates, Inc., 2011.

[76] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In F. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 25*, pages 1097–1105. Curran Associates, Inc., 2012.

[77] C. Laben. and B. Brower. Process for enhancing the spatial resolution of multispectral imagery using pan-sharpening, 2000. U.S. Patent 6011875, 2000.

[78] J. D. Lafferty, A. McCallum, and F. C. N. Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the Eighteenth International Conference on Machine Learning*, ICML '01, pages 282–289, San Francisco, CA, USA, 2001. Morgan Kaufmann Publishers Inc.

[79] D. Laney. 3D data management: Controlling data volume, velocity, and variety. Technical report, META Group, February 2001.

[80] D. J. Lary, A. H. Alavi, A. H. Gandomi, and A. L. Walker. Machine learning in geosciences and remote sensing. *Geoscience Frontiers*, 7(1):3–10, 2016. Special Issue: Progress of Machine Learning in Geosciences.

[81] Y. LeCun, B. E. Boser, J. S. Denker, D. Henderson, R. E. Howard, W. E. Hubbard, and L. D. Jackel. Handwritten digit recognition with a back-propagation network. In D. S. Touretzky, editor, *Advances in Neural Information Processing Systems 2*, pages 396–404. Morgan-Kaufmann, 1990.

[82] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner. Gradient-based learning applied to document recognition. In *Proceedings of the IEEE*, pages 2278–2324, 1998.

[83] M. Li, S. Zang, B. Zhang, S. Li, and C. Wu. A review of remote sensing image classification techniques: The role of Spatio-contextual information. *European Journal of Remote Sensing*, 47(1):389–411, 2014.

[84] M. Lin, Q. Chen, and S. Yan. Network in network. In *International Conference on Learning Representations*, 2013.

[85] Z. C. Lipton, J. Berkowitz, and C. Elkan. A critical review of recurrent neural networks for sequence learning. *arXiv preprint arXiv:1506.00019*, 2015.

[86] X. Liu, L. Jiao, J. Zhao, J. Zhao, D. Zhang, F. Liu, S. Yang, and X. Tang. Deep multiple instance learning-based spatial-spectral classification for pan and ms imagery. *IEEE Transactions on Geoscience and Remote Sensing*, PP(99):1–13, 2017.

[87] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2015, Boston, MA, USA, June 7-12, 2015*, pages 3431–3440, 2015.

[88] D. Lu and Q. Weng. A survey of image classification methods and techniques for improving classification performance. *International Journal of Remote Sensing*, 28(5):823–870, Jan. 2007.

[89] D. Lu and Q. Weng. Extraction of urban impervious surfaces from an IKONOS image. *International Journal of Remote Sensing*, 30(5):1297–1311, 2009.

[90] L. Lymburner, P. Tan, N. Mueller, R. Thackway, M. Thankappan, A. Islam, A. Lewis, L. Randall, and U. Senarath. The national dynamic land cover dataset - technical report. `https://ecat.ga.gov.au/geonetwork/srv/eng/catalog.search?node=srv#/metadata/71069`, 2011. Accessed: 2018-09-26.

[91] L. Ma, F. Ma, Z. Ji, Q. Gu, D. Wu, J. Deng, and J. Ding. Urban land use classification using LiDAR geometric, spatial autocorrelation and lacunarity features combined with postclassification processing method. *Canadian Journal of Remote Sensing*, 41(4):334–345, 2015.

[92] Y. Ma, H. Wu, L. Wang, B. Huang, R. Ranjan, A. Zomaya, and W. Jie. Remote sensing big data computing: Challenges and opportunities. *Future Generation Computer Systems*, 51:47–60, 2015. Special Section: A Note on New Trends in Data-Aware Scheduling and Resource Provisioning in Modern HPC Systems.

[93] A. L. Maas, A. Y. Hannun, and A. Y. Ng. Rectifier nonlinearities improve neural network acoustic models. In *ICML 2013 Workshop on Deep Learning for Audio, Speech, and Language Processing*, 2013.

[94] J. Macqueen. Some methods for classification and analysis of multivariate observations. In *In 5-th Berkeley Symposium on Mathematical Statistics and Probability*, pages 281–297, 1967.

[95] E. Maggiori, Y. Tarabalka, G. Charpiat, and P. Alliez. Convolutional neural networks for large-scale remote-sensing image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(2):645–657, Feb 2017.

[96] G. Mariethoz and P. Renard. Reconstruction of Incomplete Data Sets or Images Using Direct Sampling. *Mathematical Geosciences*, 42(3):245–268, Apr. 2010.

[97] J. F. Mas and J. J. Flores. The application of artificial neural networks to the analysis of remotely sensed data. *International Journal of Remote Sensing*, 29(3):617–663, 2008.

[98] N. Mboga, C. Persello, J. Bergado, and A. Stein. Detection of informal settlements from vhr images using convolutional neural networks. *Remote Sensing*, 9(11):1106, 2017.

[99] M. Mhawej, G. Fahour, C. Abdallah, and J. Adjizian-Gerard. Towards an establishment of a wildfire risk system in a Mediterranean country. *Ecological Informatics*, 32:167–184, 2016.

[100] C. Miller, M.-A. Parisien, A. A. Ager, and M. A. Finney. Evaluating spatially explicit burn probabilities for strategic fire management planning, 2008.

[101] J. P. Minas, J. W. Hearne, and D. L. Martell. A spatial optimisation model for multi-period landscape level fuel management to mitigate wildfire impacts. *European Journal of Operational Research*, 232(2):412–422, 2014.

[102] L. Mosley. *A balanced approach to the multi-class imbalance problem*. PhD thesis, Iowa State University, Ames, Iowa, 2013.

[103] Y.-K. Mousa, P. Helmholz, and D. Belton. New dtm extraction approach from airborne images derived dsm. *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, 42(1W1):75–82, 2017.

[104] V. Mulder, S. de Bruin, M. Schaepman, and T. Mayr. The use of remote sensing in soil and terrain mapping—a review. *Geoderma*, 162(1–2):1–19, 2011.

[105] I. A. Mundo, T. Wiegand, R. Kanagaraj, and T. Kitzberger. Environmental drivers and spatial dependency in wildfire ignition patterns of northwestern Patagonia. *Journal of Environmental Management*, 123:77–87, 2013.

[106] M. D. Mura, J. A. Benediktsson, B. Waske, and L. Bruzzone. Morphological attribute profiles for the analysis of very high resolution images. *IEEE Transactions on Geoscience and Remote Sensing*, 48(10):3747–3762, Oct 2010.

[107] K. Murphy. *Machine Learning: A Probabilistic Perspective*. The MIT Press, 2012.

[108] M. H. Nami, A. Jaafari, M. Fallah, and S. Nabiuni. Spatial prediction of wildfire probability in the Hyrcanian ecoregion using evidential belief function model and GIS. *International Journal of Environmental Science and Technology*, 15(2):373–384, Feb. 2018.

[109] T. Ojala, M. Pietikainen, and T. Maenpaa. Multiresolution gray-scale and rotation invariant texture classification with local binary patterns. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 24(7):971–987, Jul 2002.

[110] S. Paisitkriangkrai, J. Sherrah, P. Janney, and A. van den Hengel. Semantic labeling of aerial and satellite imagery. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 9(7):2868–2881, 2016.

[111] M.-A. Parisien, S. Snetsinger, J. A. Greenberg, C. R. Nelson, T. Schoennagel, S. Z. Dobrowski, and M. A. Moritz. Spatial variability in wildfire probability across the western United States. *International Journal of Wildland Fire*, 21(4):313, 2012.

[112] L. Pasolli, F. Melgani, and E. Blanzieri. Gaussian process regression for estimating chlorophyll concentration in subsurface waters from remote sensing data. *IEEE Geoscience and Remote Sensing Letters*, 7(3):464–468, 2010.

[113] C. Persello and A. Stein. Deep fully convolutional networks for the detection of informal settlements in vhr images. *IEEE Geoscience and Remote Sensing Letters*, 14(12):2325–2329, Dec 2017.

[114] C. Persello, V. Tolpekin, J. R. Bergado, and R. A. de By. Delineation of agricultural fields in smallholder farms from satellite images using fully convolutional networks and combinatorial grouping. *Remote sensing of environment*, 231:1–18, 6 2019.

[115] P. H. O. Pinheiro and R. Collobert. Recurrent convolutional neural networks for scene labeling. In *Proceedings of the 31st International Conference on Machine Learning (ICML)*, 2014.

[116] B. Polyak. Some methods of speeding up the convergence of iteration methods. *USSR Computational Mathematics and Mathematical Physics*, 4(5):1–17, 1964.

[117] J. A. Richards. *Remote Sensing Digital Image Analysis*. Springer, Heidelberg, 5th edition, 2013.

[118] O. Ronneberger, P. Fischer, and T. Brox. U-net: Convolutional networks for biomedical image segmentation. In N. Navab, J. Hornegger, W. M. Wells, and A. F. Frangi, editors, *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, pages 234–241, Cham, 2015. Springer International Publishing.

[119] J. Rowley. The wisdom hierarchy: Representations of the DIKW hierarchy. *Journal of Information Science*, 33(2):163–180, Apr. 2007.

[120] D. E. Rumelhart, G. Hinton, and R. J. Williams. Learning representations by back-propagating errors. *Letters to Nature*, 323:533–536, 1986.

[121] S. A. Sader, D. Ahl, and W.-S. Liou. Accuracy of landsat-tm and GIS rule-based methods for forest wetland classification in Maine. *Remote Sensing of Environment*, 53(3):133–144, 1995.

[122] G. Sakr, I. Elhajj, and G. Mitri. Efficient forest fire occurrence prediction for developing countries using two weather parameters. *Engineering Applications of Artificial Intelligence*, 24(5):888–894, 2011.

[123] M. Salehi, M. Sahebi, and Y. Maghsoudi. Improving the accuracy of urban land cover classification using Radarsat-2 PolSAR data. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, 7(4):1394–1401, 2014.

[124] Y. Sánchez Sánchez, A. Martínez-Graña, F. Santos Francés, and M. Mateos Picado. Mapping Wildfire Ignition Probability Using Sentinel 2 and LiDAR (Jerte Valley, Cáceres, Spain). *Sensors*, 18(3):826, Mar. 2018.

[125] J. Sanyal and X. X. Lu. Application of remote sensing in flood management with special reference to Monsoon Asia: A review. *Natural Hazards*, 33(2):283–301, 2004.

[126] G. Scarpa, S. Vitale, and D. Cozzolino. Target-adaptive cnn-based pansharpening. *CoRR*, abs/1709.06054, 2017.

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

[128] J. Sherrah. Fully convolutional networks for dense semantic labelling of high-resolution aerial imagery. *arXiv preprint arXiv:1606.02585*, 2016.

[129] I. Sikder, S. Mal-Sarkar, and T. Mal. Knowledge-based risk assessment under uncertainty for species invasion. *Risk Analysis*, 26(1):239–252, 2006.

[130] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[131] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *International Conference on Learning Representations*, 2015.

[132] J. T. Springenberg and M. Riedmiller. Improving deep neural networks with probabilistic maxout units. *arXiv preprint arXiv:1312.6116*, 2013.

[133] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich. Going deeper with convolutions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–9, 2015.

[134] A. Tassetti, E. Malinverni, and M. Hahn. Texture analysis to improve supervised classification in IKONOS imagery. In *International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences - ISPRS Archives*, volume 38, pages 245–250, 2010.

[135] The State of Victoria. Fire history records of fires primarily on public land. `https://www.data.vic.gov.au/data/dataset/fire-history-records-of-fires-primarily-on-public-land`, 1992. Accessed: 2017-02-07.

[136] The State of Victoria. Road network - vicmap transport. `https://www.data.vic.gov.au/data/dataset/road-network-vicmap-transport`, 2009. Accessed: 2018-09-22.

[137] The State of Victoria. Vicmap features of interest. `https://www.data.vic.gov.au/data/dataset/vicmap-features-of-interest`, 2009. Accessed: 2018-09-22.

[138] The State of Victoria. *Code of practice for bushfire management on public land*. Dept. of Sustainability and Environment, Melbourne, 2012.

[139] M. P. Thompson and D. E. Calkin. Uncertainty and risk in wildland fire management: A review. *Journal of Environmental Management*, 92(8):1895–1909, 2011.

[140] M. P. Thompson, J. R. Haas, J. W. Gilbertson-Day, J. H. Scott, P. Langowski, E. Bowne, and D. E. Calkin. Development and application of a geospatial wildfire exposure and risk calculation tool. *Environmental Modelling and Software*, 63:61–72, 2015.

[141] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler. Efficient object localization using convolutional networks. In *computer Vision and Pattern Recognition (CVPR 2015)*. arXiv preprint arXiv:1411.4280, 2014.

[142] D. Tuia, J. Verrelst, L. Alonso, F. Pérez-Cruz, and G. Camps-Valls. Multioutput support vector regression for remote sensing biophysical parameter estimation. *IEEE Geoscience and Remote Sensing Letters*, 8(4):804–808, 2011.

[143] C. Vasilakos, K. Kalabokidis, J. Hatzopoulos, and I. Matsinos. Identifying wildland fire ignition factors through sensitivity analysis of a neural network. *Natural Hazards*, 50(1):125–143, 2009.

[144] E. Vermote. Mod09a1 modis/terra surface reflectance 8-day l3 global 500m sin grid v006. `https://lpdaac.usgs.gov/`, 2015. Accessed: 2018-07-19.

[145] M. Volpi and V. Ferrari. Structured prediction for urban scene semantic segmentation with geographic context. In *2015 Joint Urban Remote Sensing Event (JURSE)*, pages 1–4, March 2015.

[146] M. Volpi and D. Tuia. Dense semantic labeling of subdecimeter resolution images with convolutional neural networks. *IEEE Transactions on Geoscience and Remote Sensing*, 55(2):881–893, 2017.

[147] M. Voltersen, C. Berger, S. Hese, and C. Schmullius. Object-based land cover mapping and comprehensive feature calculation for an automated derivation of urban structure types at block level. *Remote Sensing of Environment*, 154:192–201, 2014.

[148] L. Wan, M. Zeiler, S. Zhang, Y. L. Cun, and R. Fergus. Regularization of neural networks using dropconnect. In *Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, pages 1058–1066, 2013.

[149] Z. Wan, S. Hook, and G. Hulley. Mod11a2 modis/terra land surface temperature/emissivity 8-day l3 global 1km sin grid v006. `https://lpdaac.usgs.gov/`, 2015. Accessed: 2018-10-22.

[150] L. Wang, X. Huang, C. Zheng, and Y. Zhang. A markov random field integrating spectral dissimilarity and class co-occurrence dependency for remote sensing image classification optimization. *ISPRS Journal of Photogrammetry and Remote Sensing*, 128(Supplement C):223–239, 2017.

[151] C. H. Wickramasinghe, S. Jones, K. Reinke, and L. Wallace. Development of a multi-spatial resolution approach to the surveillance of active fire lines using himawari-8. *Remote Sensing*, 8(11), 2016.

[152] B. Xu, N. Wang, T. Chen, and M. Li. Empirical evaluation of rectified activations in convolutional network. In *Deep Learning Workshop, ICML 2015*, 2015.

[153] M. Yebra, P. E. Dennison, E. Chuvieco, D. R. no, P. Zylstra, E. R. Hunt, F. M. Danson, Y. Qi, and S. Jurdao. A global review of remote sensing of live fuel moisture content for fire danger assessment: Moving towards operational products. *Remote Sensing of Environment*, 136:455–468, 2013.

[154] H. Yildirim, M. Özdemir, M. Güre, B. Ugurlu, and M. Özel. Monitoring the after-effects of forest fires by satellite data. In *RAST 2005 - Proceedings of 2nd International Conference on Recent Advances in Space Technologies*, pages 626–629, 2005.

[155] D. Yu, H. Wang, P. Chen, and Z. Wei. *Mixed Pooling for Convolutional Neural Networks*, pages 364–375. Springer International Publishing, Cham, 2014.

[156] F. Yuan, K. E. Sawaya, B. C. Loeffelholz, and M. E. Bauer. Land cover classification and change analysis of the twin cities (Minnesota) metropolitan area by multitemporal Landsat remote sensing. *Remote Sensing of Environment*, 98(2–3):317–328, 2005.

[157] M. D. Zeiler. Adadelta: An adaptive learning rate method. *CoRR*, abs/1212.5701, 2012.

[158] M. D. Zeiler and R. Fergus. Stochastic pooling for regularization of deep convolutional neural networks. *arXiv preprint arXiv:1301.3557*, 2013.

[159] M. D. Zeiler and R. Fergus. Visualizing and understanding convolutional networks. In *European Conference on Computer Vision*, pages 818–833. Springer International Publishing, 2014.

[160] M. D. Zeiler, M. Ranzato, R. Monga, M. Z. Mao, K. Yang, Q. V. Le, P. Nguyen, A. W. Senior, V. Vanhoucke, J. Dean, and G. E. Hinton. On rectified linear units for speech processing. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2013, Vancouver, BC, Canada, May 26-31, 2013*, pages 3517–3521, 2013.

[161] C. Zhang, I. Sargent, X. Pan, H. Li, A. Gardiner, J. Hare, and P. M. Atkinson. An object-based convolutional neural network (ocnn) for urban land use classification. *Remote Sensing of Environment*, 216:57–70, 2018.

[162] Q. Zhang and J. Wang. A rule-based urban land use inferring method for fine-resolution multispectral imagery. *Canadian Journal of Remote Sensing*, 29(1):1–13, 2003.

[163] R. Zhang and D. Zhu. Study of land cover classification based on knowledge rules using high-resolution remote sensing images. *Expert Systems with Applications*, 38(4):3647–3652, 2011.

[164] W. Zhao, L. Jiao, W. Ma, J. Zhao, J. Zhao, H. Liu, X. Cao, and S. Yang. Superpixel-based multiple local cnn for panchromatic and multispectral image classification. *IEEE Transactions on Geoscience and Remote Sensing*, 55(7):4141–4156, 2017.

Summary

A

Prediction plays a crucial role in mapping urban land cover, land use, and wildfire risks. Data-related challenges concern the spectral confusion of the classes of interest, noisy classification results, highly abstracted class definitions, and the large amount of spatial, temporal, structural, and typological variation. Standard methods tackle these issues by introducing pre- and post-prediction steps such as image fusion, feature extraction, and label regularization. These additional steps are often approximate and detached from the prediction task at hand, making them sub-optimal. This research explores end-to-end predictive models for several remote sensing applications, integrating independent additional steps within a single prediction pipeline.

The first objective addresses image fusion in the context of urban land cover classification. I present a single-stage framework embedding feature extraction and image fusion in a multiresolution convolutional network, called *FuseNet*. The network matches the resolution of the panchromatic and multispectral bands in a very high resolution (VHR) image using convolutional layers with corresponding downsampling and upsampling. I compared FuseNet with the use of separate processing steps for image fusion, such as pansharpening and resampling through interpolation. Results show quantitative improvements on the accuracy of the land cover classification results when using FuseNet.

The second objective addresses label regularization in the context of urban land cover classification. I propose a novel end-to-end classification model integrating image fusion, feature extraction, and label regularization within a recurrent multiresolution convolutional network, called *ReuseNet*. ReuseNet extends FuseNet by incorporating contextual label information via recurrent connections. This extension is similar to a standard post-classification label regularization step. I designed experiments on land cover classification where I compared ReuseNet with using separate processing steps for both image fusion and map regularization. Experimental results show both quantitative

and qualitative improvements on the classified land cover maps produced by ReuseNet.

The third objective addresses predicting abstracted urban land use classes. Features extracted from land cover maps are helpful on performing land use classification. Such prior information can be incorporated in the design of a deep learning based land use classifier by applying a multitask learning setup, i.e. simultaneously solving a land use and a land cover classification task. I explored a fully convolutional multitask network to classify urban land use from VHR imagery. I experimented with three different setups: a standard network only predicting the land use class of each pixel in the image, a multitask network concatenating the land use and land cover class labels in the same output layer, and a multitask network accepting as an input the land cover that was predicted by a subpart of the network, concatenated to the original input image patches. Comparing the three against a standard random forest classifier, I found that the two convolutional multitask networks outperform the other two classifiers by at least 30% in the average F1-score.

The fourth objective deals with integrating a big set of geodata to produce daily maps of the probability of wildfire burn. I designed a deep fully convolutional network, called *AllConvNet*, to produce daily maps of the probability of a wildfire burn over the next 7 days in Victoria, Australia for the period of 2006–2017. Fifteen factors that were extracted from six different datasets and resulted into 29 quantitative features, were selected as input. I compared AllConvNet with three baseline methods: *SegNet*, multilayer perceptron, and logistic regression. AllConvNet outperformed the three baseline methods in four of the six quantitative metrics considered. Total precipitation, lightning flash density, and land surface temperature were consistently highly weighted by all models while terrain aspect components, wind direction components, certain land cover classes (such as crop field and woodland), and distance from power lines were lowly weighted.

In summary, this thesis presents end-to-end predictive models for three different remote sensing applications: urban land cover, urban land use, and wildfire prediction. These models demonstrate that standard methods adding independent pre- and post-prediction steps can be integrated into a single end-to-end framework that streamlines the prediction task. The urban land cover and land use maps can be used for regular updating of geospatial database layers, while the wildfire probability maps can either directly serve as a proxy measure for wildfire risk or may serve as an input for probabilistically quantifying wildfire risk.

Samenvatting

Het maken van voorspellingen speelt een cruciale rol bij het in kaart brengen van stedelijke landbedekking, landgebruik en de risico's op natuurbranden. Uitdagingen die gerelateerd zijn aan de gegevens hebben betrekking op de spectrale vermenging van de betreffende klassen, onduidelijke classificatieresultaten, abstracte definities van klassen en sterke ruimtelijke, temporele, structurele en typologische variatie. Standaardmethoden pakken deze problemen aan door pre- en post-voorspellingsstappen te introduceren, zoals Image-fusie, feature-extractie en label-regularisatie. Deze extra stappen zijn vaak enkel een benadering en staan los van de voorspellingstaak, waardoor ze niet optimaal zijn. Dit onderzoek verkent end-to-end voorspellende modellen voor verschillende toepassingen van remote sensing, waarbij onafhankelijke aanvullende stappen worden geïntegreerd binnen één voorspellingspijplijn.

De eerste doelstelling betreft image-fusie in de context van de classificatie van stedelijke landbedekking. Ik presenteer een één-staps kader genaamd FuseNet, dat functie-extractie en beeldfusie in een multiresoluut convolutioneel netwerk combineert. Het netwerk opereert op het niveau van panchromatische en multispectrale banden in een zeer hoge resolutie (VHR) afbeelding en gebruikt convolutie lagen met bijbehorende downsampling en upsampling. Ik heb FuseNet vergeleken met het gebruik van afzonderlijke bewerkingsstappen voor image-fusie, zoals pansharpening en resampling door middel van interpolatie. De resultaten laten kwantitatieve verbeteringen zien in de nauwkeurigheid van de classificatieresultaten voor landbedekking bij gebruik van FuseNet.

De tweede doelstelling betreft de regularisatie van labels in de context van de classificatie van stedelijke landbedekking. Ik stel een nieuw end-to-end classificatiemodel voor dat image-fusie, feature-extractie en label-regularisatie integreert binnen een recurrent multiresoluut convolutie netwerk, genaamd ReuseNet. ReuseNet in een uitbreiding van FuseNet door contextuele label-informatie op te nemen. Deze uitbreiding is vergelijkbaar met een standaard regularisatiestap

tijdens een postclassificatie van de labels via terugkerende verbindingen. Ik heb experimenten ontworpen voor de classificatie van landbedekking waarbij ik varianten van ReuseNet vergeleken heb met het gebruik van afzonderlijke verwerkingsstappen voor zowel image-fusie als kaartregularisatie. Experimentele resultaten laten zowel kwantitatieve als kwalitatieve verbeteringen zien op de geclassificeerde landbedekkingskaarten die via ReuseNet zijn geproduceerd.

De derde doelstelling betreft het voorspellen van geabstraheerde klassen van stedelijk landgebruik. Kenmerken die uit landbedekkingskaarten zijn gehaald, zijn nuttig bij het uitvoeren van classificatie van landgebruik. Dergelijke voorinformatie kan worden opgenomen in het ontwerp van een classificator van landgebruik die is gebaseerd op deep learning door een multitask-leeropstelling toe te passen, d.w.z. het gelijktijdig oplossen van een classificatietaak voor landgebruik en landbedekking. Ik heb een volledig convolutioneel multitasknetwerk onderzocht om stedelijk landgebruik te classificeren op basis van VHR beelden. Ik heb geëxperimenteerd met drie verschillende netwerken: een standaardnetwerk dat alleen de landgebruiksklasse van elke pixel in de afbeelding voorspelt, een multitask-netwerk dat de labels voor landgebruik- en landbedekkingsklassen koppelt in dezelfde uitvoerlaag, en een multitasknetwerk dat als invoer de landbedekking, voorspeld door een onderdeel van het netwerk, koppelt met de oorspronkelijke beelden in de invoer. Door deze drie te vergelijken met een standaard random forest-classificatie, ontdekte ik dat de twee convolutionele multitask-netwerken minstens 30

De vierde doelstelling betreft het integreren van een grote reeks geodata om dagelijkse kaarten te maken van de kans op bosbranden. Ik heb een diep volledig convolutioneel netwerk ontworpen, genaamd AllConvNet, om dagelijkse kaarten te maken van de waarschijnlijkheid van een bosbrand gedurende de komende 7 dagen in Victoria, Australië voor de periode 2006–2017. Vijftien factoren die uit zes verschillende datasets werden gehaald en resulteerden in 29 kwantitatieve kenmerken, werden als input geselecteerd. Ik vergeleek AllConvNet met drie basismethoden: SegNet, een perceptron bestaande uit meerdere lagen en logistische regressie. AllConvNet presteerde beter dan de drie basismethoden in vier van de zes onderzochte kwantitatieve statistieken. Totale neerslag, dichtheid van bliksems en temperatuur van de landoppervlakte werden consequent zwaar gewogen binnen alle modellen, terwijl componenten van terreinaspecten, componenten van de windrichting, bepaalde landbedekkingsklassen (zoals akkerbouw en bos) en afstand tot hoogspanningsleidingen lager werden gewogen.

Samenvattend presenteert dit proefschrift end-to-end voorspellende modellen voor drie verschillende remote sensing toepassingen: stedelijke landbedekking, stedelijk landgebruik en voorspelling van natuurbranden. De modellen laten zien

dat standaardmethoden die onafhankelijke pre- en post-voorspellingsstappen toevoegen, kunnen worden geïntegreerd in een enkel end-to-end kader dat de voorspellingstaak stroomlijnt. De kaarten voor stedelijke landbedekking en landgebruik kunnen worden gebruikt voor het regelmatig bijwerken van geospatiale lagen in een database, terwijl de kansenkaarten voor natuurbranden ofwel direct kunnen dienen als een proxy-maat voor het risico op natuurbranden of als input voor een kanstheoretische kaartering van de risico's op natuurbranden.

Authors Biography

B

John Ray Bergado received a BSc degree (magna cum laude) in Geodetic Engineering from the University of the Philippines (Diliman, Philippines) in 2013. He then finished an MSc degree (cum laude) in Geoinformation Science and Earth Observation specializing in Geoinformatics at the University of Twente (Enschede, The Netherlands) in 2016. In April 2016, he started as a PhD candidate in the Earth Observation Science Department, Faculty of Geo-information Sciene and Earth Observation (Faculty ITC), University of Twente, the Netherlands.

The research has produced the following publications aside from this manuscript:

J. R. Bergado, C. Persello, and A. Stein. FuseNet: End-to-end multispectral VHR image fusion and classification. 2018 IEEE International Geoscience and Remote Sensing Symposium, IGARSS 2018 - Proceedings, pp. 2091-2094, Jul 2018.

J. R. Bergado, C. Persello, and A. Stein. Recurrent multiresolution convolutional networks for vhr image classification. IEEE Transactions on Geoscience and Remote Sensing, 56(11):6361–6374, Nov 2018.

J. R. Bergado, J. R. Bergado, C. Persello, and A. Stein. Land use classification using deep multitask networks. ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences, XLIII-B3-2020:17–21, Aug 2020.

J. R. Bergado, C. Persello, K. Reinke, and A. Stein. Predicting Wildfire Burns from Big Geodata using Deep Learning (submitted for review).