Land Classification Using Deep Neural Networks Applied to Satellite Imagery Combined with Ground-Level Images

Luís Ricardo Fonseca Agostinho Freixinho

Thesis to obtain the Master of Science Degree in

Information Systems and Software Engineering

Supervisors:  Prof. Bruno Emanuel Da Graça Martins
              Prof. João Carlos Gomes Moura Pires

Examination Committee

Chairperson: Prof. José Carlos Martins Delgado
Supervisor: Prof. Bruno Emanuel Da Graça Martins
Member of the Committee: Prof. Jacinto Paulo Simões Estima

May 2019
Dedicated to my family, specially my father and mother, and my friends for supporting me more than ever to accomplish this milestone.
Acknowledgments

I would like to thank my advisor Bruno Martins, for the patience he had with me during this rough and challenging period of my life.

I want to thank my friends Natalino Correia, Inês Percheiro, Guilherme Malcato, Mariana Martins and Ricardo Campos for all the support they gave me so that I could accomplish this milestone.

I would like to thank my family once again. My parents did everything in their power so that I could be the person that I am today and I hope finishing this phase will make them more proud of me.
Resumo

Com o aumento exponencial de imagens georreferenciadas de alta qualidade, acerca do nosso planeta, disponíveis na internet, em conjunto com o avanço e melhoria dos últimos anos a nível computacional e algorítmico, a tarefa de classificação de terreno torna-se a partida mais fidedigna. Isto promove que, quando aplicadas técnicas de aprendizagem profunda a estas imagens, se obtenham melhores e mais precisos resultados.

Neste momento, para além de se utilizar exclusivamente estas imagens de alta resolução para treinar modelos de aprendizagem profunda, é possível utilizar em conjunto modelos de elevação de terreno para que se obtenha uma maior precisão a nível das áreas visíveis de uma fotografia, de forma a que haja uma melhor classificação do terreno.

O mapeamento de actividades humanas associadas a terreno ou minimização de catástrofes naturais são duas possíveis aplicações (de muitas) de um Sistema de Informação Geográfica (SIG) que certamente serão benéficas para o nosso conhecimento geográfico.

Este trabalho apresenta um estudo novo em relação à viabilidade de aplicar redes de aprendizagem profunda em conjunto com técnicas de análise da visibilidade a imagens de satélite e imagens ao nível do chão para classificar terreno, quer de épocas atuais em que existem imagens de satélite para auxiliar, como em épocas passadas em que não há registos de imagens de satélite de terreno.

Palavras-chave: Calibração de Camera, Rede Neural Convolucional, Modelos de Elevação, Classificação de terreno, Análise espacial, Análise de visibilidade
Abstract

With the exponential growth of georeferenced labeled and high-detailed imagery information of our world available on the web, combined with both computational power and the algorithms improvements from the last years, the land classification task becomes more reliable from the start, and thus it is possible to achieve sharper results when applying deep learning techniques to that imagery.

Besides using exclusively high resolution data to train deep learning models, we can now apply it to terrain elevation models in order to have a more precise information about the visibility areas of a picture to better classify the land.

Mapping the multiple sources of human activity involved in land or minimizing natural hazards, are two of many applications of a Geographic Information System (GIS), and our geographic knowledge discovery benefits from having this kind of information.

This work advances a new study on the feasibility of applying deep neural networks and visibility analysis techniques to a combination of satellite imagery and ground-level images to classify land, either from nowadays (that contain satellite imagery) and also from past periods of time, when satellite imagery was not available but only ground-level photographs.

Keywords: Camera Calibration, Convolutional Neural Network, Elevation Models, Land Cover, Spatial analysis, Visibility analysis
Contents

Acknowledgments .......................................................... v
Resumo ................................................................. vii
Abstract ............................................................... ix
List of Tables .......................................................... xiii
List of Figures .......................................................... xv

1 Introduction ......................................................... 1
  1.1 Thesis Proposal ............................................... 2
  1.2 Contributions ............................................... 3
  1.3 Thesis Outline ............................................... 3

2 Concepts and Related Work ..................................... 5
  2.1 Fundamental Concepts ...................................... 5
      2.1.1 Deep Learning for Image Classification ............... 5
      2.1.2 Visibility analysis .................................. 12
  2.2 Related Work ................................................. 14
      2.2.1 Deep Learning Methods for Image Classification .... 14
      2.2.2 Terrain Classification with Georeferenced Multimedia Contents 19
      2.2.3 Camera Calibration .................................. 26
  2.3 Summary ...................................................... 28

3 Mapping Land Coverage with Ground Level and Satellite Imagery ........................................... 29
  3.1 Process ....................................................... 29
  3.2 Visibility Analysis Technique .............................. 31
  3.3 Deep Learning Applied to Land Classification .......... 33
  3.4 Implementation Overview .................................. 35

4 Results ............................................................ 37
  4.1 Datasets and Data Processing ............................... 37
4.2 Visibility Analysis Tests ................................................. 40
4.3 Land Classification Model Evaluation ................................. 43
  4.3.1 Results for the Model Pre-Trained with Historical Photos from SF (1938) 46
  4.3.2 Results for the model pre-trained with Flickr photos from SF ............. 48

5 Conclusions .................................................................. 52
  5.1 Achievements .................................................................. 53
  5.2 Future Work .................................................................. 53

Bibliography .................................................................. 53
List of Tables

4.1 Distribution of the LCLU classifications among the raster pixels of both San Francisco and New York in 1938 and 2005 ........................................... 39
4.2 Distribution of the Land Coverage classifications among the raster pixels of both San Francisco and New York in 1938 and 2005, after grouping classes ........ 40
4.3 Viewshed Analysis applied to the 100NN pixels for each SF dataset .............. 43
4.4 Impact of the visibility analysis done over the 1355 pixels over the generated dataset, 10 most nearby pictures for SF ........................................... 44
4.5 Model Pre-trained with historical photographs from SF. .............................. 46
4.6 Model Pre-trained with flickr photographs from SF. ................................. 49
List of Figures

2.1 Perceptron structure using the Unit Step Activation Function. ............... 6
2.2 Convolution process. ................................................................. 8
2.3 Convolution Neural Network simplified representation. ......................... 8
2.4 Convolution Neural Network stack of layers. .................................... 9
2.5 Recurrent Neural Network structure. $U, V, W$ - parameters; $i_n, s_n, o_n$ - input, hidden state and output at time $n$ respectively; $A$ - a NN chunk. ..................... 11
2.6 Viewshed of $p$ in a certain elevation terrain model for a certain radius. .... 13
2.7 AlexNet’s architecture configuration. ............................................. 15
2.8 Configuration of the VGG architecture with 16 layers model. ................ 16
3.1 Example of a DEM of San Francisco Area ...................................... 31
3.2 Scaling process ................................................................. 32
3.3 Relevant observer points .................................................... 32
3.4 Viewshed Analysis. In this image, the yellow dot represents the observer point while the green area represents the visible area of that observer point ..... 33
3.5 Representation of the classification model implemented ...................... 35
3.6 Representation of the whole process implemented ............................. 36
4.1 USGS EROS Rasters visualization from SF and NY areas from 1938 and 2005 40
4.2 USGS EROS Rasters visualization from SF and NY areas from 1938 and 2005 41
4.3 Digital Surface Model withing the SF and NY boundaries, before and after the upscale ................................................................. 41
4.4 Example of images from OldNYC, OldSF and two from Fickr ............... 42
4.5 Viewshed calculated for one of the pixels of the DSM (represented as yellow) and the 10 most nearby historical photos (represented as purple). To note that, different photos might be taken in the exact same coordinate. ............... 42
4.6 Average Distance of the 10 most nearby pictures from every pixel for each imagery dataset using each location ................................. 44
4.7 Geographical distribution of the photographs on the four different "5 most nearby pictures" created datasets. From left to right, SF historical, SF flickr, NY historical and NY flickr.

4.8 Example of the output (right image) after using the deep-koalarization adaptation model (pre-trained with ImageNet) to colorize an historical photo (left image).

4.9 Top: Land classification distribution comparison between the Real NY 1938 data and the predicted data by feeding the developed model, pre-trained with sequences of historical photographs of SF from OldSF, with historical and historical colorized photos of NY; Bottom: Land classification Class distribution comparison between the Real NY 1938 data and the predicted data by feeding the developed model, pre-trained with sequences of historical photographs of SF from OldSF, with Flickr photos of NY.

4.10 Land Classification Maps generated with the developed model, regarding the pre-train using OldSF geotagged ground-level photographs.

4.11 Land Classification Maps generated with the developed model, regarding the pre-train using nowadays geotagged ground-level photographs from Flickr.

4.12 Top: Land classification distribution comparison between the Real NY 1938 data and the predicted data by feeding the developed model, pre-trained with sequences of photographs of SF from Flickr, with historical and historical colorized photos of NY; Bottom: Land classification distribution comparison between the Real NY 1938 data and the predicted data by feeding the developed model, pre-trained with sequences of photographs of SF from Flickr, with Flickr photos of NY.
Chapter 1

Introduction

Land classification models provide fundamental information in multiple spatial planning contexts. This specific computer vision task can be split in two main classifications, Land cover and Land use. Both are demanding classification tasks, aiming to describe the components that exist on the surface of the planet and the functional activities of a certain area respectively.

Multiple studies, some using deep learning techniques (Zhu and Newsam, 2015), have explored efficient techniques to produce these types of classification, through supervised learning, using either aerial imagery or ground-level photographs, however the combination of both sources to create this kind of classification systems using convolutional neural networks is still a poorly explored area.

Now, more than ever, we have access to plenty land imagery data that has been provided, everyday during the past years, by millions of people that keep feeding the web’s insatiable appetite for content by posting photos on social networks, images that were captured by satellites and, from now on, the aerial shots from drones. All these media resources have been used as contributors to expand our geographic knowledge discovery. With historic content, such as geotagged photographs from last century and the one before, appearing on the internet freely available it is now possible to expand our past geographic knowledge too.

The amount of photographs posted every minute on the internet contribute to a valuable source of Volunteered Geographic Information (VGI), potentially containing timely geographic information, that can help to give another perspective of the land that aerial images cannot obtain, and assigning the citizen to an important role in land mapping tasks, like (Foody et al., 2017) considers. Working with the provided content from platforms containing millions of photographs, such as Flickr and others do, it is possible to train deep neural networks models.

After the ImageNet Large-Scale Visual Recognition Challenge 2012, Russakovsky et al. (2015), winning entry belonging to Krizhevsky et al. (2012), the approach to problems related
to image classification usually are solved by feeding a state-of-the-art deep Convolutional Neural Network (CNN) model with a large data source of information in order to retrieve a classification label. Multiple algorithms have been developed to interpret high-level properties of images and produce estimations, such Workman et al. (2017a) estimated for natural beauty using CNNs trained with ground-level photographs.

With visibility analysis composed by viewshed techniques, that incorporate terrain elevation models and camera calibration methods, is now possible to assign photographs to specific geolocations autonomously, in order to generate geographic maps under a desired classification, which recently slowly started to be under sight of some research studies.

With these available resources and known technological mechanisms, now it is possible to create a system that uses geotagged photographs either from the past and the present (available in social networks and VGI) combined with satellite imagery (made accessible by reliable institutions that collect these ground truth information) to explore geographical knowledge either from the past or from the present.

In this case specially, this new novel approach, besides trying to map land coverage classification from the current years in order to compare with the ground truth of land coverage maps from the present, it is testing the feasibility to use photographs from the 19th and early 20th Century (combined to the trained model developed) to generate land coverage maps of periods of time when resources such as aerial photographs or satellite imagery were not a reality, which might help exploring this type of geographical knowledge of some epochs of our history that until now we did not have knowledge about. The code used for the development of this approach can be found on my Github¹.

1.1 Thesis Proposal

This thesis aims to provide another possible solution to the demanding classification task of land classification, that usually requires a lot of costly resources in order to be reliable, exploring a different technique.

Since, nowadays, there are a lot of images containing possible descriptors of land (either from the present and from the past) freely available on the internet, the key idea behind it, is to perform a geographically land classification, through an automatic system, that combines ground-level photographs and satellite imagery, applying viewshed techniques and deep neural networks.

In the end, the goal is to understand how well it can classify a new geographic location using

¹https://github.com/luisfreixinho/Thesis-Land-Coverage-Classification
available imagery regarding that place and also if it is possible to classify land, using historical photographs, of past periods of time, in which there was no access to some resources that today we take for granted, such as satellite imagery.

1.2 Contributions

The main contribution of this M.Sc thesis is the implementation of a system that mechanizes, through deep learning, mapping land coverage classification, relative to periods of time in which satellite imagery or aerial photographs were not available to aid the land classification task, using ground-base photographs (from nowadays and historical) and satellite imagery from today.

1.3 Thesis Outline

This dissertation, from this section on, is structured as follows. Chapter 2 overviews the state of the art works and fundamental concepts that served as foundations for this thesis. Chapter 3 clarifies the whole process behind the implementation of the developed classification model and its details. Chapter 4 reports regarding the evaluation of the model while on Chapter 5 compiles both the conclusions obtained from this project and possible future work to improve it.
Chapter 2

Concepts and Related Work

On this chapter, fundamental concepts and related work regarding the scope of this dissertation are presented and overviewed.

Section 2.1 overviews fundamental concepts about Deep Learning (Section 2.1.1) and Visibility Analysis (Section 2.1.2). Section 2.2 presents related developed work in the area of Deep Learning regarding Image Classification (Section 2.2.1), Terrain Classification using Georeferenced Multimedia Content (Section 2.2.2) and Camera Calibration Techniques (Section 2.2.3). On Section 2.3 it is presented a summary of the whole section.

2.1 Fundamental Concepts

Spatial analysis can be referred as a geographical subject defined by explaining patterns of human behavior and its spatial expression. This analysis allows the creation of multiple models to map information about specific features, mostly socioeconomic (such as population, urbanization, pollution, among others).

In order to create models that map this information, it is necessary to understand the concepts related to computer vision, such as image classification, and also related to elevation models. On this section it is possible to find a brief overview of some deep learning methods to image classification and visibility analysis techniques.

2.1.1 Deep Learning for Image Classification

The term image classification is related to one of the core tasks of computer vision, which has become ubiquitous in our society recently, and it represents a problem defined by assigning a single label (or a distribution over multiple labels), from a certain amount of classes, to a collection of image-format, or other kind of spatial data input, enabling the possibility for computers.
to visually analyze environments. The computational process of solving an image classification problem is challenging for an autonomous system but it resembles human beings’ innate techniques, characterized by recognizing patterns based on prior knowledge and afterwards applying those patterns to classify a different image, unknown until that point, that are used daily to solve trivial problems of this nature. While we see lines, contours and objects, computers see matrices of numbers.

Computers automatically learn by recognizing complicated patterns and produce smart decisions based on a certain input through artificial neural networks (ANN). These are usually structured in layers that consist in multiple wired simple processing nodes containing an activation function that, based on their input (resulting from the output of others nodes), can be activated which consequently defines their output.

The most simple form of an ANN structure is the one which produces linear binary classifications based on a 3 step process. A node, upon receiving a signal, multiplies it by a weight value (assigned to the input where the signal arrived). After all the signals of each input being multiplied, they are all summed up to produce a single value to which will be added an offset. Note that at the start all the weights and this offset are random for each node/neuron which are slightly modified after each learning iteration in order that the next result is a bit closer to the desired output. In the end, the sum is led to an activation function (transfer function) which produces the output signal (see Figure 2.1).

To train the model, after each instance it is calculated the difference between the output and the actual output in order to modify, based on a certain learning rate, the offset and the weight values. The whole process is divided in epochs, that consist in the iterations over all instances of the input, and ends under a certain rule, such as when the difference between the output and
the actual output becomes 0.

Based on this structure, the multilayer perceptron (MLP) emerged, consisting in three layers of nodes, i.e. input, hidden and output layer, with every non-input utilizing a non-linear activation, differentiating this ANN from the original perceptron due to its ability to distinguish non-linearly separable information. The training process of an MLP is achieved through the backpropagation algorithm.

For computers, an image representation is composed by a two-dimensional brightness array of intensity values from 0 to 255. This array can lead to three intensity matrices, in case we work with an RGB image, for each color channel.

In the beginning, solving an image classification problem would pass through a two-step approach, where the first was the responsible part for extracting useful features that would lead to the second part, where an image would be classified based on the features extracted before, creating a very strong accuracy dependency link between step one and step two. Nevertheless, recent developments in deep learning approaches have improved the performance of these visual recognition systems.

Deep Learning can be defined as a machine learning technique that accomplishes feature extraction and transformation as well as pattern analysis/recognition and classification by processing multiple layers of abstraction (nonlinear information). A deep-learning network, DNN, distinguishes from a common single-hidden-layer neural because of its depth (i.e. the number of layers through which data passes in the pattern recognition process).

Considered one of the most popular DNN architectures, the Convolutional Neural Networks (CNNs) which can be interpreted as Neural Networks that share their parameters across the space, are built over a chain of individual layers, each of them achieving different results. This chain is characterized by each step receiving feature maps and sending to the next step a new feature map until the last step, the fully connected layer, that will produce a map to a class of probability for a certain given feature map as input.

A major component that defines this architecture are the convolutions, which can be illustrated by Figure 2.2: Receiving a certain image, this input can be represented by its width, height and depth (the RGB color channels, mentioned before). From this representation of the image, we will consider a small portion/chunk, with a defined size, called filter and we will slide (shifting the filter by a certain amount of pixels, i.e. stride) it throughout the whole image, which will run several neural networks for each area it passes returning $k$ outputs for each chunk. In the end we obtain a different image with different dimensions and more important a different depth, in this case $k$-size. A ConvNet will be then a Deep Neural Network that has stacks of
Figure 2.2: Convolution process.

Figure 2.3: Convolution Neural Network simplified representation.

these convolutions instead of having stacks of matrix multiply layers.

The objective of a convolutional network is to progressively apply convolutions in order to reduce the spatial dimension of the input while increasing the depth (Figure 2.3) so in the end we remain having only parameters that map to features of the initial image in order to apply a classifier afterwards.

Most of the times, each step in a ConvNet can be represented by a group of layers containing a convolutional layer, a rectified linear units layer and a pooling layer. A brief description of each layer and some other essential components can be found below:

- **Filters**: are represented by a vector of weights with which we convolve the input. Sometimes referred as a kernel or neuron, it provides a measure for how close a chunk of input
Figure 2.4: Convolution Neural Network stack of layers.

(note that the depth dimension of the chunk has to be the same as the depth dimension of the input) resembles a feature. This feature that is derived from the data through the learning algorithm and not defined manually.

- **Convolution layer:** using filters (mentioned before), it will compute the output of neurons that are connected to local chunks in the input, each computing a dot product between their weights and a small region they are connected to in the input volume. The process of doing this for all local chunks is called convolving process, and the amount by which the filter shifts is the stride. The output, which is a feature/activation map, will keep the depth value of the amount of filters that are used on this step.

- **Rectified Linear Units (ReLU) layer:** usually, ReLU layers are applied after Conv Layers with the particularity of adding nonlinear properties to the system without affecting significantly the accuracy. It applies an element-wise activation function, \( f(x) = \max(0, x) \), to all values in the input volume, thresholding the activation at zero.

- **Pooling layer:** it performs a down-sampling operation (usually maxpooling) along the spatial width and height dimensions, resulting in a reduction of parameters or weights (which will reduce the computational work) and attenuating overfitting.

- **Fully-connected (FC) layer:** will compute the class scores, outputting a dimensional vector of number of classes size, for a given input resulted from a previous conv, relu or pooling layer. This outcome is a representation of which features most correlate to a particular class.

In Figure 2.4 it is possible to see how a ConvNet works, independently being shallow or deep (i.e. having few layers like older models or thousands of layers like more recent models). In the beginning there is an identification of low-level features (e.g. color detection or boundaries) and gradually the model identifies higher-level features, which usually are very refined textures related to the parts of the image that stimulates the most for some desired feature.
Another type of architecture is the Recurrent Neural Network, RNN, a powerful model for sequential data modeling and feature extraction that brings the learning persistence concept to neural networks. These networks are called recurrent due to the fact they have a “memory” based on the information computed so far in the process. This memory expands from performing the same task for each element of the input sequence in loop, which will result in an output that was influenced by the previous computations of the process. Their structure is represented on Figure 2.5. Training a RNN is similar to training a traditional Neural Network since it also uses a backpropagation algorithm, although the parameters are shared through all steps and the gradient at each output depends also the previous time steps. In other words, $s_n$, which is dependent and calculated based on the previous hidden state using a nonlinearity function such as ReLU, $s_n = f(Ui_n + Ws_{n-1})$, is considered the memory of the network, which will take effect on the predicted output, $o_n = \text{softmax}(Vs_n)$.

The most common type of RNN are Long Short Term Memory, LSTM. This type of RNN, comes to solve problems related to the vanishing gradient and exploding gradient effects that happens in a standard RNN architecture and makes it unable to handle long-term dependencies, introducing a ”memory unit” in the network to remember information for a long time.

LSTMs, since are RNN, are structured as chain of repeating modules of NNs, although instead of having a single neural network layer like a normal RNN on each module, it has four. The core idea of LSTM is the existent cell state that flows through the entire chain carrying information that is modified structures called gates controlling which new memory component vector $x_t$ and hidden state component vector $h_t$ information that is added at a $t$ time.

To decide which information is discarded from a cell state is introduced the ”forget gate” layer, that is defined by the sigmoid function $f_t = \sigma(W_f \times [h_{t-1}, x_t] + b_f)$, by looking at $h_{t-1}$ and $x_t$ and outputting a value, between 0 and 1 depending, if the information is to be discarded or not, for each value in the cell state $C_{t-1}$.

To decide which new information is going to be added to the cell state, an ”input gate” is responsible to define which values will be updated. This gate is defined by the sigmoid function $i_t = \sigma(W_i \times [h_{t-1}, x_t] + b_i)$. The new values that could be added to the cell state are defined in a vector format, by a tanh layer, and are defined as $Q_t = \sigma(W_C \times [h_{t-1}, x_t] + b_C)$. To update the old memory cell ($C_{t-1}$)) into the new cell state ($C_t$), it is calculated the function $C_t = f_t \times C_{t-1} + i_t \times Q_t$.

The sigmoid function $O_t$, output gate, is responsible to decide what is the output, which will depend on the cell state. To decide what parts of the cell state that is going to be outputted is applied a sigmoid layer, $o_t = \sigma(W_o[h_{t-1}, x_t] + b_o)$, then the cell state passes through an hyperbolic
Figure 2.5: Recurrent Neural Network structure. \( U, V, W \) - parameters; \( i_n, s_n, o_n \) - input, hidden state and output at time \( n \) respectively; \( A \) - a NN chunk.

tangent, \( \tanh \), activation function to only output the desired parts of it, \( h_t = o_t \times \tanh(C_t) \). A variant and simpler variant of this architecture is the Gated Recurrent Unit (GRU), that mainly has an update gate \( z \), composed by a combination of the forget and input gates of a LSTM, that will define how much previous information needs to be pushed along the future, and a reset gate \( r \), which defines how the new input should combine with the previous memory. Besides this combination and other minor changes, the cell state and hidden state are merged to the hidden state \( h \). The equations that compose the GRU are the following:

\[
\begin{align*}
z_t &= \sigma(W_x x_t + U_z h_{t-1} + b_z) & (2.1) \\
r_t &= \sigma(W_r x_t + U_r h_{t-1} + b_r) & (2.2) \\
h'_t &= \tanh(W_x + r_t \odot U h_{t-1} + b_h) & (2.3) \\
h_t &= z_t \odot h_{t-1} + (1 - z_t) \odot h_t & (2.4)
\end{align*}
\]

\( z_t \) is used to calculate the update gate value for a time step \( t \), \( r_t \) calculates how much of the past information is to forget at time step \( t \), \( h'_t \) calculates the relevant information from the past by calculating the Hadamard product (element-wise product of two matrices) between the reset gate \( r_t \) and \( U h_{t-1} \) and applying the nonlinear activation function \( \tanh \). The \( h_t \) function is needed to calculate the vector holding information regarding the current unit to pass through the network. To do so, it computes which data is needed to collect from memory at time step \( t \), \( h'_t \) and the data coming from the previous steps \( h_{(t-1)} \)
2.1.2 Visibility analysis

Since the proposed task relies on the combination of satellite imagery and ground-level images, it is important to understand which techniques can be applied to the image data in order to be able to map the geo-spatial information of the visible terrain of a ground-level picture into a grid map of regions. This can be translated as a visibility problem.

This visibility problem can be derived in two different problems. The first one, the visibility queries, results in the possibility or not, for two given points (an observer point and a target point) in space for them to be visible by each other, i.e. the observer point being able to visibly reach the target point due to the lack of obstacles in between them. The second one, viewsheds, tries to discover which parts (set of points \( q \)) of the terrain \( T \) are visible for a single (or multiple) viewpoint \( p \).

\[
\text{viewshed}(p) = \{q \in T \mid q \text{ is visible from } p\} \tag{2.5}
\]

Being able to extract the composition of visible points for a given source point is exactly what was needed for being able to map a photo according to the map grid, although this process can be very CPU consuming, so it is mandatory to apply an efficient algorithm to do so. In Floriani and Magillo (2003), the author presents several computations methods of terrain elevation models in order to calculate the visibility areas of several regions.

The representation of the terrain data usually can be done in two forms, by a Triangulated Irregular Network (TIN) - which its surface is divided in multiple planar triangles and the elevation of a point \( p \) on that surface is a bi-linear interpolation of the elevations of the three vertices of that compose the triangle where \( p \) is inserted - or by a Raster Digital Elevation Model (DEM) - consisting in a simple grid matrix of elevations, usually, equally spaced. None of these representations is better than the other one. The DEM can achieve sharper analysis results of the terrain elevation with a simpler data structure, although it requires more memory space than TINs. In both cases, in order to \( p_1 \) and \( p_2 \) (surface points) being mutually visible, the line segment that connects both point has to lie strictly above the surface.

When computing the viewshed of a viewpoint (Figure 2.6), it is also common to take in consideration the radius of interest, \( r \), of that point, in order to acknowledge the visibility limitations, which can be formally defined as:

\[
\text{viewshed}(p, r) = \{q \in T \mid \text{distance}(p, q) < r \land \text{isVisible}(p, q) = True\} \tag{2.6}
\]

Besides viewsheds, there are two other visibility structures related to the horizon of a viewpoint, local and global horizons. The local horizons are described as the loci of points visible from the
viewpoint that block the view of the immediately located beyond points while the global horizon is the farthest point visible in every direction of the viewpoint.

\[
\text{localhorizons}(p) = \{ q \in T \mid \text{isVisible}(p, q) = True \land \exists x \forall y, x \neq q \in T \\
\text{such that } q \in \overline{px} \text{ and } y \in \overline{qx}, \text{ isVisible}(p, y) = True \} 
\] (2.7)

\[
\text{globalhorizon}(p) = \{ q \in T \mid \text{isVisible}(p, q) = True \land \forall x, q \in \overline{px}, \text{ isVisible}(p, x) = False \\
\text{isVisible}(p, x) = False \} 
\] (2.8)

In order to compute multiple viewsheds for multiple points on a given terrain, although exists multiple algorithms to execute this computation it is necessary to take in consideration the computational workload associated to that same algorithm. In Andrade et al. (2011) it is mention their algorithm External Memory Viewshed Computation, EMVS, as a possible optimized solution.

The EMVS runs on DEM Terrains Stored in the External Memory and has as base a method that for a terrain \( T \) represented by a \( n \times n \) matrix \( M \), a point \( p \in T \), a radius of interest \( r \) and a height \( h \) computes the viewshed of \( p \) within a distance \( r \) from \( p \) by positioning the observer’s point over the point \( p \) at an height of \( h \) and creating a square in the plane \( z = 0 \) with \( 2r \times 2r \) dimensions centered on \( p \). With this implemented, the algorithm radial sweeps using lines of sight starting from \( p \) and going through all the cells \( c \) inside the square’s area in order to verify if the cells belonging to the lines of sight are visible from \( p \). A position \( q \in T \) is visible from \( p \) if \( \overline{pq} \) does not pass through any position which its height > \( q \)’s height.

These lines of sight are defined by connecting \( p \) to each point that resides in the square
perimeter, while the other points inside the square are set as "not visible". For each LOS \( l \), the algorithm starts in \( p \) and sets \( l \)'s slope = \(-\infty\) and will update this value every time it reaches a cell that produces a LOS with a bigger slope. This happens if \( l \)'s height on the position of a cell \( c \) is lower than the \( c \)'s height. \( c \) is marked as visible and \( l \) has its height modified to value corresponding to \( c \)'s height.

The problem with this algorithm, that motivated Andrade et al. (2011) to improve it, is the fact that the cells are accessed in a sequence defined by the radial sweep what would make this algorithm unacceptable and not viable to apply on big terrains since it would require random access to the file which would require a lot execution time.

The idea to solve this problem consists in generating a list, composed by multiple tuples of cells and their identifier index (about when should the cell be processed), with the terrain’s points sorted by the processing order. This way, when calculating the viewshed, the algorithm runs over the list without needing to random access the file. This list is stored in external memory and is accessed by a special library, which allows efficient manipulation of external data.

2.2 Related Work

Several recent studies try to retrieve similar computer vision results to the human vision ones. Some of these works focus on the development of the most commonly used CNN architectures, machine learning methods to classify land and visibility analysis techniques. Relevant related work about these topics are presented on this section.

On Section 2.2.1, there is overview of the developments achieved regarding the Deep Learning to solve Image Classification Tasks, specially detailing some of the most used models nowadays for this machine learning area of study. Section 2.2.2 reveals studies associated with the classification of terrain using georeferenced multimedia contents and some of them use deep learning methods. Last Section (2.2.3) considers camera calibration techniques that integrate the visibility analysis field, regarding what is visible within a picture and also several techniques to reflect on and consider to integrate on future work.

2.2.1 Deep Learning Methods for Image Classification

Convolutional Neural Networks have been used in image classification tasks for almost 40 years now. The first successful application of a convolutional network was presented by LeCun et al. (1998), with LeNet architecture, used mainly for character recognition tasks. This architecture receives a 32x32 pixels sized images that would pass through three convolutional layers, with
Figure 2.7: AlexNet’s architecture configuration.

6, 16 and 120 feature maps respectively, two subsampling operations, responsible for computation complexity reduction, with 6 and 16 feature maps respectively, in succession and a fully connected layer (connected to the last convolutional layer) containing 84 units and one output layer. This last uses the Euclidean Radial Basis Function (RBF), which computes the distance between the input vector and the parameter vector. The RBF value increases the further the input vector is from the parameter vector.

In 2012, comes up a strong reappearance of this topic promoted by the fast-paced evolution of computational power, improved algorithms and the availability of larger datasets that led to AlexNet (Supervision), by Krizhevsky et al. (2012), which was a huge success when applied to large-scale imagery, by outperforming all previous non deep learning based models by a significant margin. This architecture is based on a convolutional layer followed by pooling layer and a normalization and again one more convolutional and pooling layers and normalization. After this normalization follows a few more conv layers, a pooling layer, and then several fully connected layers, making a total of five convolutional layers and three fully connected layers.

The first convolutional layer, in this architecture, will filter received ImageNet inputs at a size 227 by 227 by 3 images with the appliance of 96 11x11 filters with a stride of 4, resulting in an output with a 55x55x96 volume. The following layer, which is a pooling layer, applies 3x3 filters with a stride of 2, reducing the spatial dimensions of the input to 27x27x96. This keeps on repeating for each layer specification until the architecture achieves 1000 neurons the last fully connected layer (as represented on 2.7), representing the 1000 ImageNet classes.

An interesting point about this architecture is that AlexNet was trained on old GPUs that only had three GBs of memory, which made it impossible to fit the entire network on one GPU, so the network had to be spread across two GPUs (each would have half of the neurons, or half of the feature maps) and they would communicate only at specific layers, more specifically on
AlexNet has been used a lot from then until a couple years ago, although multiple efforts have been made to enhance the SuperVision’s architecture and achieve convolutional networks state-of-the-art results on classification tasks. One example of these tries was presented by Simonyan and Zisserman (2014), that addressed the depth aspect on ConvNets by fixing all the other parameters and adjusting the depth by adding more convolutional layers, up to 19 weight layers (after performing tests it was concluded that with 19 layers the error rate of the architecture starts to saturate), in order to understand how would it affect the neural network’s accuracy and performance in the large-scale image recognition setting.

To proceed with this evaluation, the authors described the design of their models as unusual (compared to others until then), since its compositions presents upfront two different key details - much deeper networks with much smaller filters. So, besides (in some of their VGG models) going only from the 8 typical AlexNet layers to 16 or 19 layers, they combined that with the usage of very small (the smallest possible in fact to describe the neighbourhood of a pixel), but stacked, filters on convolution layers (i.e 3x3 stacked receptive fields with stride 1 and pad 1 ) with the objective of taking advantage of the large receptive field they produce (e.g three 3x3 filters have the same receptive field as one 7x7 filter) plus the increase of non-linearity and the decrease of the amount of parameters to learn \(3 \times (3^2C^2) < (7^2C^2)\) for C channels per layer), that a single 7x7 filter would not offer. The Pooling layers perform 2x2 max-pooling without padding and 2 of stride.

Three fully-connected layers follow the stack of convolutional layers, being the first two composed by 4096 channels each, and the last one with 1000 class channels.

With this simple structure (example of VGG16 2.8) all the way through the network, these VGG models achieved 7.3% top five error on the ImageNet challenge at the year they were presented.

In terms of memory, the variant model of 19 layers requires a little bit more than the one with 16 since, in practice, it can achieve slightly better results. Compared to the AlexNet one,
the VGG16/VGG19 end up outperforming in terms of classification over the 1000 classes and localization accuracy, although they use more memory overall.

In the same year VGGNet achieves the second place in the ImageNet competition for classification, the first place went to Szegedy et al. (2015) that revealed GoogLeNet, a very deep (22 layers) and efficient, in computational terms, architecture. This was possible due to the appliance of a stack of inception modules and replacing any expensive fully connect layer in the network, which largely reduced the amount of parameters in the network.

The idea behind the inception module comes from the need of making a decision for the type of convolution that should be made at each layer. So with the inception module every convolution is used (1x1, 3x3, 5x5 convolutions, 3x3 max pooling, etc) and the model will decide by itself which one should be applied by parallel computing each convolution and concatenating depth-wise the resulting feature map before moving to the posterior layer.

Since there are multiple convolutions being produced in parallel it is expected that the computational complexity of this task to be very high, very expensive compute. To fight back this problem, the authors describe the dimensionality reduction mechanism used, by applying 1x1 convolutions to lower the dimension before the expensive convolutions, by preserving the spatial dimension and reducing the depth.

Based on this inception module, Chollet (2016), Keras library creator, suggests, in 2016, a novel deep convolutional neural network architecture where these inception modules have been replaced with depth-wise separable convolutions without non-linearities such as ReLU, Xception. This architecture is composed by 36 stacked conv layers with residual connections. On this research, the author performs evaluations comparing both Xception and Inception (Inception-V3 more precisely) models over a classification task using the ImageNet dataset to predict a single-label over 1000 possible classes and using JFT dataset to estimate multi-label classification over 17000 classes, where Xception achieved slightly better classification results when applied to the ImageNet dataset over Inception V3 and significantly better when applied to the JFT dataset.

More recently, due to the state-of-the-art performance that residual connections, advanced by He et al. (2016), in conjunction with traditional deep learning architectures were achieving, Szegedy et al. (2017) reports that the training of Inception networks can be significantly accelerated by training with residual connections, suggesting Inception V4. With Inception V4, a combination of the default Inception architecture with residual networks is proposed, by adding a connection to each inception module. After performing tests, the authors concluded that what improves the models’ training is the addition of this kind of connections instead the model’s
Howard et al. (2017) present MobileNet, a new model built over the suggestion of a factorization of standard 3x3 convolutions, into one depth-wise 3x3 convolution, followed by a point-wise convolution, which was proved more efficient. The model considers hyperparameters that choose higher accuracy or performance. This choice reflects in several advantages of these model in comparison to other state-of-the-art models, such as reduced network size, number of parameters, more fast in performance and small, low-latency convolutional neural network. It is also considered interesting for mobile applications.

Others CNNs are also considered, such as, a novel unit architecture design that was recently proposed by Hu et al. (2017) where were implemented “Squeeze-and-Excitation” blocks trying to improve the representational capacity of a network achieving state-of-the-art performance. The authors suggest this structure could be considered helpful to related fields such as network pruning for compression.

An example of a deep learning method applied to an image classification task is advanced by Workman et al. (2017a). In order to predict image scenicness and after analyzing that the natural beauty of a picture can be related to both low-level features and semantic properties, the authors used a CNN to automatically estimate the scenicness of a photography. For that, three different models were developed to try better achieving human-like results. One model was trained to predict an average single value for scenicness (between 0 and 10), another to predict a distribution and a last one they model the set of ratings for an image as a sample from a multinomial distribution. For the predicting task it was used a modified version of GoogLeNet CNN initialized with weights from another CNN pre-trained with content related to this task of classification landscape pictures. In this case, the last model was, comparatively to the human’s classification the most consistent in terms of prediction.

Another application of deep learning was the segmentation task which was used to crop the section of an image that would contribute the most for its resulting scenic value from the previous predictive task. In the end, the authors applied a cross-view technique to extend and adapt their approach to overhead imagery, based on the geotagged ground-level images used on the previous approaches, in order to be able to map the scenic values. For this exercise they gathered a Bing overhead image (with a specific setting of zoom) for each ground-level they had on the original dataset in order to cross-view training the overhead images based on the scenicness prediction of the ground-level ones. This training was done over a variant of the usual cross-view approach, Cross-View Hybrid, where for each query location of an overhead image, the prediction of its scenicness uses the features resulting from combining the overhead
image features, the $k$ closest ground-level images features and the weighted distances to between the overhead and each of the ground-level images from $k$, using a small feed-forward network. This network has three hidden layers, an hyperbolic tangent sigmoid as activation function and the weights are regularized by an L2 loss. This results in a prediction of a rating distribution belonging to a ground-level image taken at the overhead image location.

The evaluation of this prediction is done by comparing the result with the rating of the closest ground-level image(s) obtained from the first approach.

Other works apply deep learning techniques as a procedure to retrieve modifications of original images to improve the results of possible posterior computer vision tasks, as e.g detection, object classification or segmentation tasks, by colorizing gray scale images in a indistinguishably way from real color photos through per-pixel classification problems, as presented in Zhang et al. (2016) and in Larsson et al. (2012), or even by image-to-image high quality translation between multiple domains like Choi et al. (2017) show.

### 2.2.2 Terrain Classification with Georeferenced Multimedia Contents

Classifying land based on georeferenced media results from outputting an objective label to a portion of land using imagery that belongs to it, which can be challenging, due to the implicit subjectivity that such type of content can transmit. Another challenge results from the exponential growth of the available and noisy media content online. Previously, geographic knowledge discovery tasks, such as land use and, specially, land cover mapping were performed mostly with limited and exclusive content like topographic maps, satellite imagery and other overhead imagery e.g. the national land cover database, which were resource-expensive to collect due to the detailed information they had to contain.

Nowadays, with the emergence of social networks and photo sharing communities it is possible to quickly obtain an enormous quantity of data spending much less resources, that can speed up the expansion of our geographic knowledge, however most of this data are either ground level photographs or videos that were collected from multiple available online sources, which, besides having the underlying photographic factors to take in account (e.g. as orientation or viewpoint), they carry more complicated problems to deal with, so it is necessary to understand beforehand the utility and feasibility of using these kind of resources. Antoniou et al. (2016) proposed an analysis to the usefulness of using Volunteered Geographic Information (VGI) issue.

For this investigation task, three different photography repositories (Flickr, Geograph and Panoramio) were exposed to multiple analysis to understand how useful could be the information attached to the photographs, based on their usage to classifiers training, land use and land
cover (LU/LC) map classifiers and as a complement validation. During this process, the authors identified which datasets offer more utility regarding the amount and quality of available metadata about location information, date information, camera orientation, tags, titles, et al and also about the land cover identification capability of the photo. Based on the authors’ results, which revealed that, even though the absence of a formal and standardize metadata (e.g. tags) among these three photo sharing websites, a substantial amount (more than half in the analysis set) of photographs were capable to transmit useful LULC information, they concluded that there is potential using geotagged photographs in land cover usage analysis despite the existent informality in these kind of online data repositories.

In extension to the previous approach, Tracewski et al. (2017), focus on retrofitting an existing pre-trained model, Places205-AlexNet neural network (being Places the database introduced by Zhou et al. (2014)), on a variety of scene characteristics in order to understand if this feasibility can be automatically assessed by it through processing the output of the model in two ways, being the first one based on weighting (UW) the output while the second one used the output to train a decision tree (DT) on domain-specific features of interest. The output of this neural network was branched in three different groups of classification, based on the way it was extracted: (1) 205 scene category labels; (2) 102 scene attribute labels; (3) output of the last hidden layer, that hands over 4096 values (representing high-level features) that together can be described as the image’s signature. To determine if a neural network, that was not designed to attain good results at first in identifying land cover or land used, could achieve reasonable results performing that task and if its output could be easily filtered, the authors tried 4 approaches with different goals, and for each approach, were developed three models corresponding to the possible types of classification outputs previously described.

These approaches consisted in: (1) identifying human impact in a landscape by creating five new classes, according to human presence in nature, assign weight of 1 to each one of the 205 scene category and 102 attributes, previously defined, in case they represent any of these 5 classes and after an expert labelled 965 photos with one of these classes run the UW and DT methods to check if the classes outputted by the models were the same as the class labelled by the expert; (2) filtering photos by usefulness using previous criteria and decision rules applied by Antoniou et al. (2016) to give weights of 1 or 0 to relate each available scene category and attribute with zero or more of the nine land covers defined in that study to label each photo as ‘useful’, ‘maybe useful’ or ‘not useful’ based on specific thresholds; (3) identifying land cover as defined by Urban Atlas (UA) by verifying if the labels resulted from classifying photos of Paris established a correlation with the classes present in UA at the same location; (4) identifying
UA land cover classes in the surrounding areas by, using the same method as the previous one, considering the land covers identified within a certain range around photograph’s location, so that it would be possible to take in account the precision adversities of identifying land cover at a rigorous georeferenced point based on a photograph location.

Furthermore, Xing et al. (2017) explore the usability of social media data (Sina Weibo) about Points of Interest (POI) in terms of land validation by introducing a two step validation framework that uses modified decision trees. The first step belongs to the POI classification process which considered both semantic meaning and spatial heterogeneity, which, due to the fact of the different distribution of POIs in urban and non-urban zones, the shortest distances between the POIs and both roads and villages were considered as factors in the modified DT model. Afterwards, using a majority vote algorithm, based on the number of POI types, with geographic locations and the classification results from the first step, a data transformation is applied in order to produce a raster form of the data that can identify land cover type. In the end is performed an artificial surface validation using common pixel comparison and confusion matrix methods to validate the accuracy of the classified data in comparison to the ground truth of Beijing.

In Zhang et al. (2017), is explored the urban land use classification by a parcel-based method with a Random Forest classifier using Google Street View (GSV) Images, high resolution orthoimagery and light detection and ranging data. These three sources of data were used to produce thirteen input training features for a Random Forest classifier, being the most important the parcel size, maximum of building areas, percentage of total building area, average of normalized difference vegetation index (NDVI), and the standard deviation of NDVI. The four parcel features derived from the GSV’s images were automatically detected and related to the length of possible textual information extracted from the image (from different FOVs) and were chosen to integrate this model due to the difficulty to distinguish between class labels that have a lot in common of the remaining 9 features, for instance, mixed residential & commercial buildings, single-family houses and multifamily residential buildings. This way, residential & commercial buildings, that usually contain shop signs, can be distinguished from the other two, that normally do not have present these textual clues.

After applying these features to the train the Random Forest to classify the parcels into 7 possible labels (Single-family house, Multi-family residential building, Public facility & institution, Mixed residential & commercial building, Commercial & industrial building, Park & open space and Parking facility) there were done evaluations to understand how the usage of the GSV-derived parcel features would influence the accuracy of the classification and the model’s
accuracy. The main conclusions of the results were: (i) parcel-based urban land-use classification with remote-sense data and random forests can accurately classify single-family houses and relatively accurate multi-family residential buildings, mixed residential & commercial buildings, commercial & industrial buildings, and park & open space; and the (ii) GSV-derived features improve significantly the accuracy level of classification of certain labels.

Following the previous conclusions, in Zhu and Newsam (2015) work is presented a method to map land use of a campus, which made it possible to generate a ground truth out of it manually in order to compare against their results, based on Flickr photographs. Regarding the intrinsic problems of using a data collaborative photo collection, the authors advance with techniques in order to reduce geolocation errors among other used for the rest of classification system.

Their system’s workflow can be described by: (1) gathering pictures from Flickr located within the campus range; (2) applying a Geo-Filtering with Shape Files in order to generate a more precise map in terms of land-use elements that are present in the campus, resulting in very geo-informative map. This will also remove noise from the dataset, i.e removing pictures that are not located in none of those regions that they specifically wanted to analyze; (3) a semi-supervised training data augmentation process, enhancing by increasing and balancing, in an efficient way, the training set that initially had an uneven spatial distribution of Photos among regions. This augmentation process starts by randomly selecting 20% of the photos belonging to each category in the map as part of the training and the remaining 80% to the test set. Afterwards, to the extent of the training set expansion, it was made a keyword-based search on Flickr in order to get approximately 3000 related photos for each category; (4) applying the training data straightforward to a pre-trained, with 7 million labeled images, CNN model in order to obtain high-level semantic features; (5) on the final step, the authors compared two possible methods to land-use classify their training data. The first method involved in training a single eight-way linear SVM classifier with the training data and then applying the test data, while on the second one they introduced a new classification step before to identify and separate the training data into indoor and outdoor photography and only afterwards applying, dependently of their indoor/outdoor status, to one of two eight-way land-use classifiers. The second model obtained better results than the first one, nonetheless both models’ results demonstrate that is feasible to use high-level image features extracted using CNNs from geolocated ground-level images from a collaborative image dataset to obtain great results when it is not possible to land use classify from an overhead vantage point images.

A related approach is discussed by Leung and Newsam (2015), where the authors explore other methods to evaluate and produce map land cover maps, based on a binary classification of
developed and undeveloped region, using two large collections of georeferenced data sources that differ to each other due to their initial purpose of collecting data, i.e. being Flickr and Geograph the data-sources used by the authors, the first one’s images were not necessarily contributed with the purpose of the second one, of cataloging the Earth’s surface.

In the analysis, the authors consider four different image features. Three visual features, namely color histogram, edge histogram and gist descriptors and a textual feature. After a brief study on the textual information attached to each image, immediately the authors understood that it would be a challenge to label an image merely based on the long-sentence based descriptions given by the users, so, they ended up inferring the images’ labels after applying multiple natural language processes that resulted in a transformation of the initial description in multiple usable terms. Due to the spareness of terms in the dictionary which would not be sufficient to assign a class to an image, the authors considered dividing the terrain in 1 x 1 km tiles and group the photographs that belong to each tile and from that, assign a label to the tiles (instead of the images) using the resulting terms of the images of each tile.

Two different classification systems are then considered. One that, using a weakly supervised technique to train SVMs, outputs a fraction of development result for each image, based on the amount of developed images in the same tile divided by the amount of images in the same tile, which is used to produce a binary label for that same image comparing this value to a threshold, in order to be considered developed or undeveloped. The other system is tile-based, i.e the SVMs label the tiles directly, doing an average of the features from all the images belonging to that tile which results in a binary classification of developed/undeveloped tile.

After comparing both systems to their chosen ground-truth, the authors could conclude that it is possible to produce automatically these land cover maps or other discriminating between land use classes tasks using collaborative collections of photos’ features. Another important judgment was that the results obtained by using the textual features were very influenced by photographers’ intent, showing that the Geograph dataset, a collection of geographically representative photographs, obtains a better classification result than Flickr.

Srivastava et al. (2018) use Google Street View (GSV) imagery, which is considered ground-based, to perform the classification task claiming that, due to the regular updates this dataset gets, it helps updating the produced land classification maps with lower costs. It combines this imagery with the OSM information, regarding the urban-objects on the map, to perform the classification using a Siamese Network (Bromley et al., 1993) that combines the feature vectors resulting from feeding CNN with multiple pictures, from different points of view of the different urban-objects present in that location. This way, allowed Srivastava et al. (2018) to achieve
great results on fine-grained landuse characterization.

Using both satellite imagery and geotagged ground-level photographs as input, Workman et al. (2017b) suggest a end-to-end novel DNN architecture to estimate geospatial functions (such as population density, land cover, etc). The architecture implemented, for each ground-level image, \((G_i, l_i)\) that belongs to a \(l_i\) location, retrieves the features using a CNN (in this case VGG-16 initialized with weights for Place), \(f_g(G_i)\), and for each pixel location of the aerial image there is an interpolation using Nadaraya–Watson kernel regression regarding the nearest ground-level images of that location, \(G_l\), in order to accomplish a dense feature map of \(H \times W \times 51\) dimension that characterize both appearance and distributional information of those ground-level images. For the aerial images, using the convolution layers \(\{1_{1-2}, 2_{1-2}, 3_{1-3}, 4_{1-3}, 5_{1-3}\}\) from the VGG-16 CNN model and reducing the dimensionality of the outputted feature maps, they fuse the ground-level feature map to them by applying an average pooling with a kernel size of \(6 \times 6\) and a stride of \(2\), reducing that same feature map, and then concatenating the channels dimension with the aerial image feature map at the layer \(3_3\), considered by the authors, a layer that offers good trade-off between computational cost and expressiveness.

To estimate the geospatial function, \(F(\ell(p))\) (being \(\ell(p)\) described as location of a pixel), both feature maps, previously detailed, are re-sized to \(H \times W\) using bilinear interpolation and the hypercolumn features from the conv layers \(\{1_2, 2_2, 3_3, 4_3, 5_3\}\) plus the ground-level feature map are extracted, resulting in a 1043 of length hypercolumn feature. This hypercolumn feature is finally passed to a small multilayer perceptron, consisting in 3 layers with size of \(512, 512\) and \(K\) (number of outputs for the task) and intermediate layers using a leaky ReLU activation function, will estimate the geospatial function.

A novel element tested by the authors was the adaptive, spatially varying interpolation method for constructing the ground-level image feature map based on features extracted from the overhead images instead of just the uniform kernel. After comparing the model with the two kernel bandwidth variants against 4 baselines models, the authors conclude that their approach achieves better results at resolving spatial boundaries than a ground-level images only model and at estimating features, that are difficult to process from just aerial imagery. This novel architecture, besides learning to extract optimal features and parameters for the fusion process, can also use other sparse measurements such as videos.

Rather than just using only photographs like the works reviewed so far, other works purpose models where deep learning is applied to classify and help to produce map geographic information based on videos. Examples of that are advanced by Murdock et al. (2015), that instead of purposing to solve a land cover task, the authors estimate a weather property, clouds’ coverage,
by combining geotagged and timestamped video data, retrieved from public accessible ground-
level webcams in USA, with satellite imagery to produce related cloud maps, and Zhu et al.
(2017b) that firstly performed spatio-temporal mapping of human activity using visual content
of geotagged videos from a social network, in this case Youtube.

In Zhu et al. (2017b) they faced the challenging task of developing an effective and efficient
video analysis framework by modeling an hidden two-stream networks in the interest to prove
that georeferenced videos can be used to map human activity in large scale. During their
experiments they chose 8 classes related to sports and 2 more to demonstrate how flexible a
model like this can be. The baseline for their work is a two-stream network approach, which
decomposes a video into its spatial and temporal streams by working with both still frames of
the video and optical flow information. Over both streams are applied distinct Convolutional
Networks recognition streams that in the end combine their results by averaging their prediction
scores.

In more detail the authors refer that, due the need of pre-computing optical flow on-the-
fly to train the temporal CNN, such computationally expensive task (crucial for a real time
application) could not be computed using the previous method so it is presented a model where
the temporal stream CNN is stacked under a MotionNet CNN, introduced by Zhu et al. (2017a),
in order to retrieve real time activity recognition by generating motion fields between frames
to estimate posterior frames by minimizing the pixelwise error between the estimation and
the original frame, and combined with a spatial stream VGG16 CNN to keep the hidden two-
stream network structure. Their obtained classifying results, when compared to another existent
generic video analysis framework, C3D (which is pre-trained with sports data), are much better,
specially in classes related to activities that contain more motion. One of the main conclusions,
after using a model based on visual content of geotagged videos is that, even though tags and
titles can help decoding visuals, visual data can transmit much more reliable information about
a scene than a dozen words subject to ambiguity and imprecision, such as a simple word like
football.

Moreover, Murdock et al. (2015) propose an extension to an existent framework that models
images sequences of dynamic textures/objects to predict and building large-scale cloud maps
using ground up images, by defining a hierarchical model that captures the dynamics textures
integrated within a Kalman filter, so that it would be possible to fuse noisy and sparse measure-
ments from regression models done over the data obtained from the webcams.
2.2.3 Camera Calibration

Camera calibration is a fundamental computer vision process that estimates the intrinsics parameters, responsible to represent a transformation projection from 3D camera's coordinates into 2D image coordinates, extrinsics parameters, that reproduces the transformation from the 3D world coordinates into the 3D camera coordinates, and distortion coefficient parameters of a camera. The estimation of these parameters, among other purposes, may serve to determine the location of a camera within a scene it captures.

These kind of camera calibration techniques are very useful when we are working with media obtained from photo sharing websites or similar content repositories, since it is not possible to calibrate using standard methods.

Workman et al. (2015a) explores this area, of estimating a camera property, by applying a deep CNN in order to assess focal length of a picture based on inputting raw pixel intensities, motivated by the fact of few images from photo-sharing collections containing specific calibration objects necessary to apply existing single-image calibration methods and by the fact that sometimes precise estimating focal length, even though not required, can be useful in many problem settings, which can be calculated easily due to advances in CNNs.

In order to estimate the focal length, the authors assume a simplified pinhole camera model and consequently the projection function can be specified as $\lambda p = KP = diag([f, f, 1])P$, where $f$ is the focal length, $P$ a world point and $K$ the camera intrinsic parameters. To obtain the $f$, it is necessary to estimate the horizontal FOV, given by $H_\theta = 2\tan^{-1}(w/2f)$, being $w$ the width of the image, which maps one-to-one with $f$. The approach of the authors passes through an adaptation of the AlexNet architecture to a regression problem where the final FC layer outputs a single node corresponding to the Horizontal FOV, the DeepFocal architecture, and train the network with transfer learning technique.

Considering images can be oriented in different ways, the essayists, propose two approaches at the training phase. One ignores the ratios and orientations and resizes all images to a default size, while in the other approach accordingly to the input image’s orientation, a specific network for that type of orientation (landscape or portrait) will receive it as training data. To test both approaches, they first look up for the impact of transfer learning starting from different network weights using two pre-trained AlexNet networks and conclude that both object-centric and scene-centric trained features are useful FOV estimation, so for the posterior evaluation they trained the model with both of those features. About the aspect ratio they conclude that the most successful strategy was to choose the specific network according to the test data.

Comparing this DeepFocal model against three other baseline predictive models
GIST descriptors, averaging FOV values, and other work techniques based on orthogonal vanishing points it was possible to conclude that a fast model like DeepFocal, which directly estimates the focal length from raw pixels by a CNN process, outperforms several baselines and can serve as solution to many high-level vision methods that require an estimate of the focal length.

In order to estimate the geographic location, viewing direction or field of views to better understand the arrangement of the elements within an image, Zhai et al. (2016) propose a method that employs large quantities of publicly geospatial information as a database with geometric information about roads, bodies of waters, etc, that, given an image at known region of interest with elements (such as the enumerated before annotated), is used to estimate extrinsic and intrinsic camera parameters to predict the exact camera location in the photograph. To model this solution, composed by 4 dimensions $\Theta = (\text{Latitude, Longitude, Azimuth, FOV})^T$, the objective passes through estimating the probability distribution function (PDF), $f(\theta; C)$, over the camera parameters, $\Theta$, for $C$ constraints. Using a proposed scored function, which encodes camera’s prior knowledge and how geometric information links between the images and the created database, the authors, by sampling this function using a Monte Carlo Markov Chain (MCMC) based strategy can estimate $f(\theta; C)$. This technique retrieved better results in comparison to other methods, without requiring nearby ground-level imagery as is typical for most vision techniques.

Workman et al. (2015b), approach the problem of cross-view image geolocation by matching ground-level images against a database of aerial images and produce a geolocation estimation for that image, instead of using other ground-level images to infer location based on visual similarities. To accomplish this solution, the authors propose a cross-view training strategy, involving using already existing CNNs to extract meaningful features from aerial images by extracting ground-level image features and then learn to predict these features from same location aerial images, without having to specify manually semantic labels. To avoid ambiguity problems when trying to match a ground-level image location to a known geolocation of the aerial image, the aerial image feature extracting function was extended to be able to extract features at multiple spatial scales allowing ground-level image to be matched to aerial images at multiple scales. Added to this, the authors developed a new dataset (CVUSA) to fulfill their needs regarding the amount of spatial scale and number of training images.

As a baseline to evaluate the localization performance of the cross-view training approach, the authors utilized, to extract features from both aerial and ground-level images, multiple, pre-trained for different occasions, of CNN models, achieving better results those that were trained for of scene classification with the Places database (in this case AlexNet and GoogLeNet).
Given the architecture of the model used on the experiments, Places (AlexNet), and its weights, for this evaluation, the cross-view training is applied to train (using their dataset images) this model, CVPlaces, to predict the output layer features by optimizing the weights. This, a single-scale model, and a multi-scale version of this model, MCVPlaces, was considered and achieved state-of-the-art results for cross-view localization.

In terms of finding the optimal parameters to extract features from the aerial images they did not achieve the results achieved by ground-level images feature extractors. Because of that, the authors considered ground-level images as better descriptors and better suited for cross-view localization.

Another work, regarding the computer vision task of prediction of position and orientation of moving camera, is presented by Brahmbhatt et al. (2017), that describe MapNet as a deep neural network that fuses visual odometry and GPS with images to solve it.

2.3 Summary

This chapter focused on an overview of previews works and also fundamental concepts regarding this topic. In the analyzed literature is possible to understand that exist some works using photographs obtained from social networks and other sources such as Flickr and GSV to feed a deep learning model in order to produce, in some cases land classification, such as Zhu and Newsam (2015), or in other cases a different type of mapping, such as Workman et al. (2017a), showing the variety of possible results obtained from this application. Besides that, it was explored deep learning models, such as MobileNets (Howard et al., 2017) which got amazing results regarding ImageNet in comparison to, for example VGG16 and GoogleNet, achieving more or less the same accuracy with considerable less resources, and there were also considered techniques of camera calibration.

Having knowledge over the work presented in this last chapter, brings a better understanding about the possibilities and state-of-the-art models in respect of this land classification task.
Chapter 3

Mapping Land Coverage with Ground Level and Satellite Imagery

With the goal in mind of developing a novel model able to, based on the combination of geo-referenced volunteered ground-level photographs and aerial imagery, produce geographic maps depending on land classification characteristics, this land classification task can be divided in four main tasks that can be briefly described as follow: (1) collection and data filtering; (2) perform visibility analysis techniques that explore terrain elevation models; (3) create a land classification model with a large-scale geo-tagged photo collection; (4) production of the land classification maps.

This chapter details the referred method developed throughout this dissertation. Firstly, clarifying the four steps previously enumerated (Section 3.1), then elaborating about the visibility analysis techniques performed and their finality (Section 3.2), followed by detailing the machine learning algorithms used for classification leading to production of the land classification maps (Section 3.3) and finalizing with Section 3.4, which overviews the implementation as a whole.

3.1 Process

Considering the related work on this field of study, mentioned in the previous section (Section 2) the developed model can be divided in four steps in order to achieve the desired land classification and posterior map production of these classifications. The steps of the process are the following:

1. Considering a discrete raster land classification map belonging to a certain geographic location produced based on a satellite image, for each regularly-spaced pixel that exists there are collected the ten most near pictures taken from the center of it. It is associated
then to each pixel’s center coordinate a dictionary containing the land classification value and an array of those ten images with their respective information, such as the image itself, date it was taken and the precise distance to the pixel. To find these ten nearest images for each pixel of the raster it is calculated the distance, given the law of haversine, between the centroid of the pixel and the coordinate of each photograph belonging to a group of hundred nearest photos efficiently calculated using the KNN algorithm from Scikit-Learn\(^1\).

2. Alternatively to just calculate the 10 nearest photos, in this step is added the component of visibility analysis. Taking in consideration the resolution of the raster containing the land classification and the resolution of a DEM, both from the same geographically location, the one with the higher resolution is converted to the same scale of the one with lower resolution with the help of the Geospatial Data Abstraction Library, GDAL\(^2\). In this case, since the DEM has an higher resolution, while converting to the same resolution as the raster one, it is performed the average of the heights of all the pixels corresponding to the new pixel on the newest scale. After lowering the resolution of the DEM, it is performed a viewshed technique, with the available plug-in for QGIS\(^3\), Advanced Viewshed Analysis\(^4\) presented on Cuckovic (2016). During the application of this visibility analysis algorithm, pixels classified as water or without classification are not considered due to efficiency reasons.

Having, for each pixel of the raster the information of which pixels are visible from that point, it is possible to take in consideration the visibility limitations of the photographs when calculating the 10 nearby images for each pixel on the previous step.

3. Having the information regarding the land classification of each pixel and its associated 10 nearby pictures, a deep learning model (which is going to be detailed on Section 3.3) is trained. This model is a recurrent neural networks, more in concrete, a GRU, which is fed with sequences of images’ feature maps (coming from using a CNN pre-trained with ImageNet), sorted by their distance to each pixel in order to associate a sequence of images to a land classification value.

Once the model is trained, it is possible to produce land classification maps of other regions from different epochs.

4. On this last step, to produce a land classification map of a new region based on the trained

---

\(^1\)https://scikit-learn.org
\(^2\)https://www.gdal.org/
\(^3\)https://www.qgis.org
\(^4\)https://github.com/zoran-cuckovic/QGIS-visibility-analysis
model relative to step 3, it is needed to gather information for this new location. This information is collected the same way as step 1 and step 2, although, now it is relative to a new geographical area and there is no data regarding that zone classification beforehand. Having associated for each pixel in the raster a sequence of nearby images, these pixels are classified using the model that was previously trained and, using GDAL, once again, it is possible to rebuild the new raster with land classification.

3.2 Visibility Analysis Technique

Independently on the location that a photography is taken, it cannot capture the whole geographic points around within an infinite radius. To calculate the visible surface from a given observer point (in this case, location where the picture is taken) over a geographical area, there is a procedure advanced on Section 2.1.2, viewshedding.

On this project, this technique is applied to calculate the viewshed area, with the help of a DEM (Figure 3.1, that represents the elevation of the geographical area considered.

![Figure 3.1: Example of a DEM of San Francisco Area](image)

Since the land classification maps produced in the end of the whole process have a lower scale, meaning that a pixel on those maps cover a wider area than a pixel on the DEM, due to efficiency reason to calculate viewshed for each pixel, the resolution of the DEM was scaled. During this scaling process to produce the new pixels with the new height value, it is considered the average height of all the pixels of that area, as can be seen on Figure 3.2.
After upscaling the DEM, all the pixels containing a non relevant land classification are not considered for the viewshed analysis. By non relevant it is meant pixels without a valid land coverage classification or classified as open waters. After this filtering, the DEM and all the considered relevant observer points can be represented as it is displayed on Figure 3.3.

Once all the relevant observer points are found, the next step is responsible to produce the viewshed for each one of this observer points in order to understand which pixels and corresponding coordinates are visible. Doing this, the information which associates a coordinate to a list of seen coordinates becomes available in order to filter out the photographs that cannot capture that area. Figure 3.4 is visual representation of this viewshed process for one of the observer points.

To gather the ten nearby images of a pixel regarding the previous process, it was implemented, with the help of GDAL library, a searching function so that for each pixel, retrieves its viewshed raster file and considers, using the k-nearest neighbors algorithm from Scikit Learn, which are the closest 10 pictures that are a visible point in the raster, by verifying to which pixel in the raster the picture belongs to and then if it belongs to a pixel with the visible value.
3.3 Deep Learning Applied to Land Classification

With Section 2.1.1, it is possible to understand how deep learning can be applied for Image classification, on Section 2.2.1 to follow some usage of deep learning methods for image classification and on Section 2.2.2 it is presented some methods of classifying terrain using georeferenced multimedia contents in which, some of the cases, deep learning methods are applied as seen with the publications of Workman et al. (2017b) or Srivastava et al. (2018). Since this is a classifying land task that, like others, it is treated as a computer vision problem (image classification problem) and on previous works it was possible to solve similar problems applying deep learning methods and good results for this work it was implemented a land classification model using deep learning methods.

This implemented novel land classification model is a combination of two models. Firstly when receiving a list of the sequence of relevant images for each pixel of a raster, it is needed to extract high-level features from those same images. For this first task, it was implemented a CNN architecture available on the high-level python Neural Network library, Keras and it was treated as a transfer learning. This CNN architecture is the MobileNet. Since it is lightweight due to performing a single convolution on each colour channel instead of combining all the channels and flattening them, as advanced on Howard et al. (2017) by stating that in MobileNets, there is a factorization, instead of doing a standard convolution that filter and combines inputs into a new set of outputs in one step which can be seen as, instead of capturing the details regarding of only one channel it is also capturing how the details of a channel coorelate to every other channel of the input (the relationship inside a channel and between channels) . The 3x3 depthwise convolution, it applies one filter for each input colour channel of the image and then the pointwise
convolution applies a 1x1 convolution that combines the depthwise convolution outputs. By doing this factorization, it is possible to reduce both the model size and the computation power. The problem might be the accuracy, but the trade-off between accuracy and computational power was taken in consideration, and due to that it was the chosen architecture. Just a brief comparative example, the full size of a VGG16, due to its depth and the number of FC nodes (reflecting in 138 million parameters), is around 533MB while, the MobileNet having 4.2 million parameters has 17MB in size.

Like others CNN architectures that are available on Keras, the MobileNet model is also implemented and pre-trained with the one of the largest resource regarding collection of images, the ImageNet\(^5\) dataset. Once implemented the MobileNet to use the Imagenet weights it will be possible to extract the most important features of the images that exist linked to a pixel of raster. For that, all considered images are cropped to a size of 224x224, which is one of the admissible input shapes for this model. Instead of including the fully-connected layer at the top of the MobilenNet, the output feature map, with a size of 7x7x1024, will be passed to a RNN model, in this case, a Gated Recurrent Unit. A GRU, as overviewed on Section 2.1.2, is a RNN similar to the LSTM, although, the cell state and the hidden state are merged into an hidden state \(h\) that is used to transfer information, and uses only two gates. One that determines how it should combine the input that is receiving with the memory it has, reset gate, \(r\) and one that delineates how much of previous memory is considered, the update gate \(z\). Having these two gates mechanism this GRU model will be able to train them and decide what information should be passed to the output along the time.

These types of RNN, are usually implemented when it is needed to model data that exists in sequence, such as speech recognition or time series prediction for example. In this case we are modeling data as a sequence of images’ features. Each feature map, coming from the CNN, is related to an image present in a sequence (of 10 images). Each image is closer (or at the same distance) than the next one to the point they are all associated with.

A figure of the classification model can be seen on Figure 3.5. The network is then trained as a whole using adam optimization and categorical focal loss function \(^6\). The last one tries to put more training emphasis data harder to classify mostly due to having less representative elements of that class on the dataset.

Once the model is trained, it can be loaded and used to predict the classification of newer sequences of images or even produce land classification maps, by combining the results provided by the model and creating a new raster using GDAL.

\(^5\)http://www.image-net.org/
\(^6\)https://github.com/umbertogriffo/focal-loss-keras
3.4 Implementation Overview

This third chapter presented the approach for modeling the land classifier. Each one of the steps that make part of the whole classifier were detailed as some of the choices of the implementation.

Section 3.1 overviews the different steps of the process in order to comprehend the model as a whole, from the transformation of the data obtained from the rasters, the filtering step using visibility analysis over the multiple coordinates of the raster and the deep learning techniques used for land classification.

Section 3.2 details information how the visibility analysis, through viewshed, was implemented to take part in the whole system.

Section 3.3 presents the structure of the deep learning model used that was trained and used to predict new possible land classification maps.

As a visual overview of the whole process developed and implemented, Figure 3.6. To note that the step 4 represents the production of a land classification map considering a new input of sequences of images from another region. This means that the model can be trained with photos obtained from a VGI. Then that data is going through all the previous process to generate the dataset "10 most nearby pictures", regarding the coordinates of each pixel on a raster and that dataset (e.g. Flickr photos). Once trained, the model can predict relatable information, such as photos from other data source (e.g. OldSF) and then, after those photos going also through the whole process previously presented (in order to associate them to pixels on a raster) it is possible to map land based on their information.
Figure 3.6: Representation of the whole process implemented
Chapter 4

Results

This chapter reflects an explanation about all evaluation methodologies applied for the procedures regarding the whole land classification model.

The goal of the following experiments is to understand the feasibility of mapping land coverage from nowadays and the past by exploring deep learning methods to ground level georeferenced historical photos and recent photos in combination with satellite imagery. A description of the datasets and data processing that was used for the evaluation phase is presented in Section 4.1. On the posterior Section, 4.2, it is detailed how the visibility analysis was evaluated and tested. On 4.3 it is possible to observe the scenarios and evaluation methodology regarding the land classification model tests.

4.1 Datasets and Data Processing

This section presents the datasets that were used during the evaluation and implementation of the model developed for this thesis. A description and a brief analysis of them is presented in the following paragraphs.

To evaluate the whole model implemented, it was decided to test it over two geographical regions of the United States. This happens because we are testing how it is possible to apply deep learning techniques combining ground-level photos and satellite imagery, and also possible applications of it, namely applying it to classify land coverage from previous years using historical photos and historical land classification data, and there is information regarding those topics associated with the United States, more precisely with San Francisco and New York.

Being San Francisco (SF) and New York (NY) the two geographical areas of study of the following experiments, in order to have a baseline regarding the land classification we are using
datasets provided by USGS\(^1\) EROS researchers, the first one being the Modeled Historical Land Use and Land Cover for the Conterminous United States: 1938-1992\(^2\) dataset, and the second one the Conterminous United States Land Cover Projections - 1992 to 2100 dataset \(^3\). Both contain information relatively to land coverage (LC). From the first one it is used the dataset relative to 1938, which had LC information that was directly input by researchers and from the second one, the dataset relative to 2005, which was produced considering the National Land Cover Database (NLCD)\(^4\), USGS Land Cover Trends\(^5\), and US Department of Agriculture’s Census of Agriculture\(^6\) to recreate the LC classification of that time. Both these datasets are rasters, with a pixel resolution of 250 meter, that contain LCLU information, divided in 17 classes, from the whole US, so it was considered for these evaluation exclusively the areas within the bounding boxes relative to SF and NY, defined respectively by (37.860909, -123.055561, 37.686071, -122.365444), containing 255x65 pixels, and (40.912455, -74.25243, 40.500425, -73.712143), containing 199x152 pixels. Within these areas, the amount of pixels that are associated to each one of the LC classification labels can be observed on the Table 4.1 and a visual representation of these classification within the defined locations on Figure 4.1.

Although, due to the fact that some LC classes have so few representative elements it is proposed a grouped version of some of these classes. Firstly the original Class 0, 3, 4 and 5 are not considered, then Class 1 ”belongs to water” (represented with blue), Class 2 belongs to ”Urban/Developed Area” (represented with red), Class 3 ”Agriculture/Grass/Forest” (comprehending the classes 8 until 16 from the original dataset, represented with green) and finally Class 4, ”Others” (containing data relative to classes 6, 7 and 17 from the original dataset and represented as gray). On Table 4.2 it is possible to view the distribution for this modified dataset with the relative rasters representation on Figure 4.2.

Regarding the visibility task, instead of using a DEM, we used a DSM, more specifically the version 2.1 of the ALOS Global Digital Surface Model (DSM), ALOS World 3D-30m (AW3D30) dataset\(^7\), released by JAXA\(^8\), containing information in a DSM about the height above sea level divided in units of area of 1 degree latitude and longitude.

Once again, since the evaluation is within the area of SF and NY, the dataset was clipped to those areas. Since the resolution of the DSM was different from the LCLU raster, I decided to upscale the first one to fit the second one’s resolution, as explained on Section 3.2, by calculating

\(^{1}\)https://www.usgs.gov
\(^{2}\)https://www.sciencebase.gov/catalog/item/59d3c73de4b05fe04cc3d1d1
\(^{3}\)https://www.sciencebase.gov/catalog/item/5b96c2f9e4b0702d0e826f6d
\(^{4}\)https://www.usgs.gov/centers/eros/science/national-land-cover-database
\(^{5}\)https://www.usgs.gov/centers/wgsc/science/land-cover-trends
\(^{6}\)https://www.nass.usda.gov/AgCensus/
\(^{8}\)http://global.jaxa.jp/
<table>
<thead>
<tr>
<th>Class id</th>
<th>Class Name</th>
<th>NY'05</th>
<th>SF'05</th>
<th>NY'38</th>
<th>SF'38</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Nan</td>
<td>2566</td>
<td>12048</td>
<td>2566</td>
<td>12048</td>
</tr>
<tr>
<td>1</td>
<td>Open Water</td>
<td>6267</td>
<td>2111</td>
<td>6113</td>
<td>2120</td>
</tr>
<tr>
<td>2</td>
<td>Urban/Developed</td>
<td>19620</td>
<td>1760</td>
<td>14980</td>
<td>1568</td>
</tr>
<tr>
<td>3</td>
<td>Intentionally Left Blank</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>4</td>
<td>Intentionally Left Blank</td>
<td>5</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>5</td>
<td>Intentionally Left Blank</td>
<td>13</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>6</td>
<td>Mining</td>
<td>7</td>
<td>4</td>
<td>10</td>
<td>5</td>
</tr>
<tr>
<td>7</td>
<td>Barren Source</td>
<td>54</td>
<td>37</td>
<td>73</td>
<td>35</td>
</tr>
<tr>
<td>8</td>
<td>Deciduous Forest Source</td>
<td>530</td>
<td>3</td>
<td>2166</td>
<td>3</td>
</tr>
<tr>
<td>9</td>
<td>Evergreen Forest Source</td>
<td>26</td>
<td>107</td>
<td>885</td>
<td>112</td>
</tr>
<tr>
<td>10</td>
<td>Mixed Forest Source</td>
<td>515</td>
<td>77</td>
<td>1290</td>
<td>84</td>
</tr>
<tr>
<td>11</td>
<td>Grassland Source</td>
<td>0</td>
<td>306</td>
<td>0</td>
<td>476</td>
</tr>
<tr>
<td>12</td>
<td>Shrubland Source</td>
<td>0</td>
<td>118</td>
<td>0</td>
<td>111</td>
</tr>
<tr>
<td>13</td>
<td>Cultivated Cropland Source</td>
<td>8</td>
<td>0</td>
<td>640</td>
<td>0</td>
</tr>
<tr>
<td>14</td>
<td>Hay/Pasture Source</td>
<td>11</td>
<td>0</td>
<td>99</td>
<td>0</td>
</tr>
<tr>
<td>15</td>
<td>Herbaceous Wetland Source</td>
<td>520</td>
<td>4</td>
<td>631</td>
<td>3</td>
</tr>
<tr>
<td>16</td>
<td>Woody Wetland Source</td>
<td>106</td>
<td>0</td>
<td>795</td>
<td>0</td>
</tr>
<tr>
<td>17</td>
<td>Perennial Ice/Snow Source</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

| Total   | Sum of all classes         | 30248 | 16575 | 30248 | 16575 |
| Non 0   | Sum of non 0 classes       | 27682 | 4527  | 27682 | 4527  |

Table 4.1: Distribution of the LCLU classifications among the raster pixels of both San Francisco and New York in 1938 and 2005

how many pixels from the DSM were need to fit one pixel at the scale from the LCLU raster, and calculating the raster. A visual representation of this upscaling for each region can be seen on the Figure 4.3.

For the evaluation of the land classification model based on georeferenced photographs and satellite imagery, we used three datasets, that can fall into two distinct categories, being the first category characterized by containing photographs from recent years and the second one by containing historical photos.

For the first category I used the largest public multimedia collection that has ever been released so far, the Yahoo Flickr Creative Commons 100 Million (YFCC100m\(^9\)) dataset developed by Thomee et al. (2016). This dataset comprehends an overall of 48,469,829 pictures containing a geographic coordinate distributed mainly on Europe, US West Coast of US (where San Francisco e located), US East Coast (where New York is located), South Asia and Japan. Regarding the locations of the experiments, it has 792,349 pictures taken in San Francisco and 1,144,239 taken in New York.

\(^{9}\)http://webscope.sandbox.yahoo.com/catalog.php?datatype=i&did=67
Figure 4.1: USGS EROS Rasters visualization from SF and NY areas from 1938 and 2005

<table>
<thead>
<tr>
<th>Class id</th>
<th>Class Name</th>
<th>NY’05</th>
<th>SF’05</th>
<th>NY’38</th>
<th>SF’38</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Water</td>
<td>6267</td>
<td>2111</td>
<td>6113</td>
<td>2120</td>
</tr>
<tr>
<td>2</td>
<td>Urban/Developed</td>
<td>19620</td>
<td>1760</td>
<td>14980</td>
<td>1568</td>
</tr>
<tr>
<td>3</td>
<td>Forrest/Agriculture/Grass</td>
<td>1716</td>
<td>615</td>
<td>6506</td>
<td>789</td>
</tr>
<tr>
<td>4</td>
<td>Others</td>
<td>63</td>
<td>41</td>
<td>83</td>
<td>40</td>
</tr>
<tr>
<td>Total</td>
<td>Sum of all classes</td>
<td>27682</td>
<td>4527</td>
<td>27682</td>
<td>4527</td>
</tr>
</tbody>
</table>

Table 4.2: Distribution of the Land Coverage classifications among the raster pixels of both San Francisco and New York in 1938 and 2005, after grouping classes

The second category contains two datasets. The OldNYC\textsuperscript{10} and OldSF\textsuperscript{11}, which contain 39,516 and 13,257 historical (from 1850-2000) geotagged photographs for the NY area and SF area respectively.

An example of images that are in these 3 datasets can be seen on Figure 4.4.

4.2 Visibility Analysis Tests

This section introduces the experimental process, its setup and discusses the obtained results regarding the visibility analysis achieved.

The evaluation done was respectively to the changes the visibility analysis would affect while generating the dataset of the 10 most nearby pictures for each pixel (process implementation described on the Section 3.2. Just to complement the information regarding the generation of this dataset, the data that it holds, can be represented as a dictionary of all pixels/coordinates (from USGS EROS Rasters) which have associated one of four land classification labels, a set/sequence of the 10 most nearby pictures id of the chosen imagery dataset and their respective distance to the centroid of that pixel.

\textsuperscript{10}urlhttps://www.oldnyc.org/
\textsuperscript{11}http://www.oldsf.org/
This process was tested over the SF area due to the computational process that implies producing a viewshed analysis for each pixel of DSM or DEM. Since SF dataset, has six times less LC categorized pixels (4527) than NY (27682), but still has 4527 to be processed, made it being the chosen region.

In order to calculate the viewshed for 1355 pixels of the SF DSM the setup was the following, due to the configuration requirements that the QGIS viewshed plugin, used, needed:

- Observer Height (Height from which the picture was taken relative to the ground) : 1.65m (being more or less the mean of a human height)
- Target Height : 10m
- Radius : 20km (maximum radius size that the machine - 16GB Memory - where it was processed could handle)
- Earth Curvature taken in consideration

Given this configuration, the viewshed analysis was performed for a portion of SF, producing

![Figure 4.3: Digital Surface Model within the SF and NY boundaries, before and after the upscale](image)
a binary information (0= not visible, 1= visible) in relation to the pixels that were visible (or not) from 1355 tested pixels.

The viewshed is applied as a filter after calculating the 100 Nearest Neighbours pixels containing photographs to a considered pixel, as advanced on the Section 3.2, to decide, from that group of 100 pixels, where are located the 10 nearest and visible photos relatively to those 1355 tested pixels. On Figure 4.5 it is possible to visualize an example about how those 10 photos are displaced in relation to the viewshed of chosen pixel that is being evaluated.

The first calculation revealed that, for those 1355 tested pixels and their respective 100NN pixels containing photos from Flickr (which gives a total of 135500 gathered pixels) 115019 pixels were visible containing photographs and 20481 were not. Using the the SF OldSF dataset, the resulting values were 60999 visible pixels containing photographs and 74501 not visible (Table 4.3). Once having the 100NN pixels filtered down to just contain visible pixels, it was calculated the 10 most nearby pictures regarding OldSF and Flickr dataset within SF.

Afterwards it was tested how applying this viewshed technique would differentiate the choice of the 10 most nearby pictures. For this it was compared the calculation of the 10 most nearby pictures without viewshed techniques involved with the new one, using visibility analysis. From
the 1355 pixels tested, using the OldSF dataset, 765 kept having the same 10 most nearby pictures while 590 suffered a change, which means that at least one different photograph replaced a photo from the nearby 10. When using the Flickr dataset, 1220 kept the same 10 most nearby pictures and only 135 changed at least one of them.

Since it was considered a change if one of the 10 most near pictures was replaced (regarding the tested pixels), it was tabulated how many pictures were replaced and how many were kept being considered one of those 10 pictures for each one of the tested pixels (13550 in total). For the OldSF, 9073 photos kept being the same on the 10 most nearby pictures, while 4477 possibly replaced a ”not visible” picture, while for the Flickr dataset, the values were 13053 kept being the same and 497 were pictures that were actually not visible from the its associated pixel. Both these two last results are displayed on the Table 4.4.

As it can be seen from the results displayed on the Table 4.3 and Table 4.4, when applying the visibility analysis during the computation of the 10 most nearby pictures of SF using the historic/OldSF dataset there are more modifications regarding the set’s content in comparison when using the Flickr dataset. This happens due to the different distributions of the average distance of the 10 most nearby pictures on both datasets. For SF, (region tested) with Flickr dataset, the 10 most nearby pictures of a pixel are around 94% of the times less than 1km away, while using the OldSF, this value is around 47% (Figure 4.6). This information, combined to the fact that, usually, the closer a pixel \( x \), in a DSM or DEM, is to an observer point pixel \( o \), when applying a viewshed from that observer point, more probability the pixel \( x \) has to be visible from \( o \), explain these results.

### 4.3 Land Classification Model Evaluation

This section introduces the experimental process, its setup and discusses the obtained results regarding the land classification model implemented.

Considering these generated datasets of the ”10 most nearby (historical or flickr) photos” for each region, the land classification model, which is proposed and its implementation was described on Section 3.3, and posterior land maps generation are evaluated in this section.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Flickr</th>
<th>SF</th>
<th>OldSF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Visible Pixels Within the 100NN</td>
<td>115.019</td>
<td>60.999</td>
<td></td>
</tr>
<tr>
<td>Not Visible Pixels Within the 100NN</td>
<td>20.481</td>
<td>74.501</td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>135500</td>
<td>135500</td>
<td></td>
</tr>
</tbody>
</table>
Table 4.4: Impact of the visibility analysis done over the 1355 pixels over the generated dataset, 10 most nearby pictures for SF

<table>
<thead>
<tr>
<th>Dataset</th>
<th>Flickr SF</th>
<th>SF OldSF</th>
</tr>
</thead>
<tbody>
<tr>
<td>Pixels keeping the same 10 most nearby pictures</td>
<td>1220</td>
<td>765</td>
</tr>
<tr>
<td>Pixels changing the same 10 most nearby pictures</td>
<td>135</td>
<td>590</td>
</tr>
<tr>
<td>Total</td>
<td>1355</td>
<td>1355</td>
</tr>
<tr>
<td>Pictures keeping on 10 most nearby pictures sets</td>
<td>13053</td>
<td>9073</td>
</tr>
<tr>
<td>New Pictures on 10 most nearby pictures sets</td>
<td>497</td>
<td>4477</td>
</tr>
<tr>
<td>Total</td>
<td>13550</td>
<td>13550</td>
</tr>
</tbody>
</table>

Figure 4.6: Average Distance of the 10 most nearby pictures from every pixel for each imagery dataset using each location

mainly in terms of precision, recall, F1 score per class, and accuracy.

Either the training phase and the tests were conducted under the following computational conditions: CPU Intel Core i7 6700 CPU (3.4 GHz); NVIDIA GeForce GTX 980 GPU (4GB); 16GB of RAM.

In these carried out tests, two models were pre-trained. One using the information relative to the ”10 most nearby historical photos” (which as said before, can be seen as a list of pixels, their coordinates, their LC class value and information related to its 10 most nearby pictures) and the other one using the generated dataset of the ”10 most nearby flickr photos”. Both San Francisco area, which means that the training samples for these tests are composed by pixels/coordinates, land coverage classification and photos from SF.

These models were trained using batches of 3 pixels. For each pixel, and due to memory problems, they were chosen the 5 most nearby pictures (instead of 10) to be inputted, sorted by the distance to the pixel, on a MobileNet to retrieve its feature maps from the last layer, and followed by feeding a GRU with those feature maps and trained according to the LC value that the sequence was label. The training was set up with batches of 3 sequences of 5 images.
Figure 4.7: Geographical distribution of the photographs on the four different "5 most nearby pictures" created datasets. From left to right, SF historical, SF flickr, NY historical and NY flickr.

Figure 4.8: Example of the output (right image) after using the deep-koalarization adaptation model (pre-trained with ImageNet) to colorize an historical photo (left image) due to memory limitations. The distribution of the location of all the photographs present in the "5 most nearby pictures" for each location and type of photography (historical of flickr) can be seen on Figure 4.7, and it is possible to perceive that when using the Flickr dataset, the "5 most nearby pictures" are located in a wider area in both cities than when using the Historical datasets. This observation complements what is visible on Figure 4.6 since, when using the Flickr dataset, the chosen pictures cover a wider area which results in having more photographs near each pixel.

The models were trained during 30 epochs and there were considered data augmentation techniques, such as flipping horizontally the images, in order generate more possible sequences. For each one of these models, there were preformed three tests, using the following input/test data (referent to a geographically location that the model did not now about) to classify:

- Generated dataset of the 5 most nearby historical photos of New York;
- Generated dataset of the 5 most nearby historical photos colorized of New York;
- Generated dataset of the 5 most nearby flickr photos of New York.

Relatively to the test using the historical photos colorization, it is used an adaptation of the deep-koalarization\textsuperscript{12}, implemented by Nuno Ramanlal, in order to colorize the historical photos

\textsuperscript{12}\url{https://github.com/baldassarreFe/deep-koalarization}

45
Table 4.5: Model Pre-trained with historical photographs from SF.

<table>
<thead>
<tr>
<th>Testing Datasets</th>
<th>NY Hist Photos (5)</th>
<th>NY Color Hist Photos (5)</th>
<th>NY Flickr (5)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Classes</td>
<td>Precision</td>
<td>Recall</td>
<td>F1</td>
</tr>
<tr>
<td>1</td>
<td>0.20</td>
<td>0.25</td>
<td>0.22</td>
</tr>
<tr>
<td>2</td>
<td>0.54</td>
<td>0.54</td>
<td>0.54</td>
</tr>
<tr>
<td>3</td>
<td>0.21</td>
<td>0.14</td>
<td>0.17</td>
</tr>
<tr>
<td>4</td>
<td>0.00</td>
<td>0.02</td>
<td>0.01</td>
</tr>
<tr>
<td>Acc</td>
<td>0.383</td>
<td>0.45</td>
<td>0.39</td>
</tr>
</tbody>
</table>

and posterior generation of the 5 most nearby historical photos colorized of NY dataset. An example of this colorization is represented on Figure 4.8.

On Section 4.3.1 it is possible to observe the results regarding training the classification model using historical photos and how it can predict using recent old and recent photos to produce land maps. On Section 4.3.2, the approach regarding using recent photos to train a model in order to possibly reveal the classification of the past is evaluated.

### 4.3.1 Results for the Model Pre-Trained with Historical Photos from SF (1938)

This section presents the results obtained by pre-training the classification model with sequences of 5 historical photos of SF, from OldSF, in order to predict/map the land classification of NY in three ways: using sequences of 5 images regarding historical photos, colorized historical photos and also Flickr photos.

Table 4.5 reports the obtained experimental results (in decimal values) regarding the accuracy of the prediction from each test and also the precision, recall and F1-score associated to the each class (of the defined 4) prediction from each test.

The results obtained, from each one the three tests, demonstrate that the classification model obtained the best results classifying sequences of images with the class 2 (Urban/Developed) on the correct pixel, achieving the highest values for the precision, recall and F1-score. This can be explaining because of the unbalance of the classes presented on the training phase. As seen on Table 4.2, the amount of pixels labeled as Urban/Developed in comparison to the other three classes is way bigger. Not even using the data augmentation process or the usage of the focal-loss loss function in order to consider more weight for the least used classification labels, made the other classes obtain a better values according to this metrics.
Figure 4.9: Top: Land classification distribution comparison between the Real NY 1938 data and the predicted data by feeding the developed model, pre-trained with sequences of historical photographs of SF from OldSF, with historical and historical colorized photos of NY; Bottom: Land classification Class distribution comparison between the Real NY 1938 data and the predicted data by feeding the developed model, pre-trained with sequences of historical photographs of SF from OldSF, with Flickr photos of NY.

Figure 4.9 shows how the classifier distributed the land coverage labels among the pixels of the rasters. When trying to predict the land coverage map using historical photos non-colorized, it is possible to see a similar distribution of pixels falling in each of the 4 classes comparing to the baseline. When using the colorized ones, there was an increase of amount of pixels classified as developed. This resulted in a higher accuracy of the model since there were more pixels being accurately labeled with the class 2, which is the class that will cause more impact in the results due to the sample number it has associated. This, made this land map being the most accurate even though its discrepancy of pixels associated to each class compared to the baseline. When using the NY flickr photos to predict the land map, the model had problems differentiating between the class 1 and 2, most likely due to the different content the flickr and historical photos contain on themselves and their location, which resulted in bad class distribution of this prediction. flickr photos contain pictures of objects interiors and pictures of the sea and rivers, which are new to a model trained mostly with black and white/yellow photos of buildings and streets.
Nevertheless, based on these results, the whole model generated the land classification rasters presented on Figure 4.10. As it can be seen, the precision of the pixel classification follows the values presented on the Table 4.5.

### 4.3.2 Results for the model pre-trained with Flickr photos from SF

This section shows experimental results gathered by pre-training the classification model with sequences of 5 nearby photos of SF, from flickr, in order to predict the land classification of NY using sequences of 5 images (historical photos, colorized historical photos and Flickr photos).

The predicted outputted classification land maps of NY are shown on Figure 4.11 and it is visible the predominance of the colours blue and red, associated with the classes 1 and 2 respectively.

As seen on the previous Section, the same happens with the results displayed on Table 4.6 obtained from testing over the pre-trained model with flickr images of SF. The class 2 is again the class achieving the best scores (reaching a F1 score achieving 70% when classifying land coverage using colorized historical photos) comparing with the other classes, most likely the because of excessive amount of training elements associated with that label.
Although the class 2 has achieved best results, now it is possible to see that in some cases, class 3 or 4, obtained no true positives when predicting correctly a pixel with the real classifying label. The difference between the dataset of the ”5 most nearby flickr photos of SF” and the used on the previous tests, is that it considers the land coverage raster equivalent to SF 2005, which contains an even greater unbalance of weight classes, since 83.67% pixels considered on the generated ”5 most nearby flickr photos of SF” dataset fall into either class 1 or 2, due to the urbanization increase over the last century.

When predicting the land coverage maps of NY 1938 using historical photos, the model achieved slightly less accurate and precise than when using flickr photos, that achieved 55% of accuracy. This happens due to characteristics and of each type of picture in each dataset. Even colorizing historical photos, the colorization is not the same as the recent photos that exist in flickr, although it increase 1% of the accuracy of the model but it is not as precise as when using the flickr dataset to predict.

The resulting distribution of the predicted classes within the NY applying respectively historical, colorized historical and flickr images to the deep learning model developed can be found on Figure 4.12, and it shows clearly the unbalanced classification of the previous 2 classes mentioned, even though it has miss classified correctly other pixels (false positives).

When looking at the predictions done with the historical photos, the class distribution is much more unbalanced than when using the flickr, which follows the baseline distribution of its associated land map.

This model, since was trained with recent photos (Flickr San Francisco), shows how a reproduction of land classification from the past (as tested when predicting using photos of NY from OldSF and their colorized versions) can be done, even though, it could not precisely predict each pixel from the raster.
Figure 4.11: Land Classification Maps generated with the developed model, regarding the pre-train using nowadays geotagged ground-level photographs from Flickr.
Figure 4.12: Top: Land classification distribution comparison between the Real NY 1938 data and the predicted data by feeding the developed model, pre-trained with sequences of photographs of SF from Flickr, with historical and historical colorized photos of NY; Bottom: Land classification distribution comparison between the Real NY 1938 data and the predicted data by feeding the developed model, pre-trained with sequences of photographs of SF from Flickr, with Flickr photos of NY.
Chapter 5

Conclusions

As the common accessible devices that are able to take pictures, such as smartphones or digital camera, keep evolving and improving their image quality by allowing us to capture the world around us in a greater detail and resolution, combining to the fact that social networks, like Flickr, Instagram and others, are keep getting flooded with these pictures (which some time they can georeferenced), and the emergence of some projects that disclose geotagged imagery datasets from the past, it can be interesting to understand how it can ease and reduce costs of certain tasks that usually can be only accomplished using expensive resources, including money, time, equipment or even human workload, such as Land Classification.

This master thesis investigates how feasible is to apply deep learning methods, such as convolutional neural networks and recurrent neural networks, to a combination of freely available ground-level photographs from different VGI (e.g. Flickr, OldNY, OldSF) with satellite imagery (providing terrain information) to classify land, either from actual years and specially from the past, in which satellite imagery to was not available but still there were ground-level image records.

As it can be seen on the evaluation Section, the results regarding how the model, that combines satelitte imagery and ground-level photographs, can precisely classify a pixel are not the best, mainly due to the unbalance of the trained samples. One of the problem has to do with the pollution that the datasets contain.

Although it was shown that using photographs from one region to train the model to classify other using its photos is possible to achieve some reasonable results regarding the land coverage class distribution and accuracy within the tested region.

In the end, this work concluded that this novel approach that considers both satellite imagery, visibility analysis and deep learning methods, treating a pixel as a sequence of nearby pictures to it, is still a demanding task, although it is possible to achieve some reasonable results even
using VGI datasets without its content filtered, deep learning models that are used on mobiles (MobileNet). With some improvements either regarding computational problems or in terms of data filtering in relation to training samples it is possible to accomplish good results while predicting land classification from the present or the past.

5.1 Achievements

The biggest achievement of this thesis was the implementation of a novel approach, that considers deep learning methods and combines two different types of images, Ground-level photographs (obtained from VGI and datasets such as OldSF) and satellite imagery to classify land coverage. Specially the possibility of the model to classify land from the past, when it was not possible to visualize the terrain from satellites, is the biggest achievement of this work.

It also considers techniques, such as visibility analysis that could improve the results regarding this classification task.

5.2 Future Work

Regarding the results obtained, there are several possible ways of improvement that can possibly raise the accuracy of the applied models in classifying precisely each pixel within a land classification raster. As future work, the model can be improved and continued by addressing the following points:

- On the initial phase of data processing, the photos should be submitted over a filtering process, in order to consider only photos revealing land classification features (specially regarding the flickr dataset that contains images about everything);

- Build the processed dataset of nearby photos for each pixel in the raster considering also a temporal distance, in order to precisely attribute to a sequence of images from a given year the exact classification of that year;

- Train and test the model under a more computational powerful machine, allowing a bigger sequence of nearby images to be considered for each pixel;

- Even though it requires more memory, consider DEM and Land Classification rasters with smaller scale, in order to precisely collect the most nearby photographs that truly correspond to a pixel.
Bibliography


