DEVELOPMENT OF A DEEP LEARNING ASSISTED WEBGIS FRAMEWORK FOR UPDATING BUILDING DATABASES

BİNA VERİTABANLARININ GÜNCELLENMESİ İÇİN DERİN ÖĞRENME DESTEKLİ WEBCBS ÇATISI GELİŞTİRİLMESİ

RECEP CAN

PROF. DR. ALİ ÖZGÜN OK
Supervisor

Submitted to
Graduate School of Science and Engineering of Hacettepe University
as a Partial Fulfillment to the Requirements
for the Award of the Degree Master of Science
in Geomatics Engineering

2022

ABSTRACT

DEVELOPMENT OF A DEEP LEARNING ASSISTED WEBGIS FRAMEWORK FOR UPDATING BUILDING DATABASES

Recep CAN

Master of Science, Department of Geomatics Engineering
Supervisor: Prof. Dr. Ali Özgün Ok
June 2022, 72 pages

The diversity of the methods for geodata collection has increased significantly in recent years. The need for automated approaches for updating geodatabases has also increased in parallel to this development. In addition to the novel machine learning (ML) methods, the contributions of non-professionals and volunteers are immensely required to achieve this goal. Geographical Information Systems (GIS) on the web and on mobile devices provide the required tools and methods for utilizing geoinformation (GI) contributed by individuals of all backgrounds. Thanks to the increasing attention on the volunteered geographic information (VGI) approaches and the Citizen Science (CitSci) projects, the contributions of non-professionals to participatory GIS efforts can be utilized gradually. On the other hand,to improve the accuracy of data obtained by volunteers, new algorithms and platforms can help with semi-automatic GI extraction. By incorporating novel artificial intelligence (AI) methods, such as the deep learning (DL) algorithms, into WebGIS, the semi-automatic GI extraction task can be facilitated. As a result, volunteers with limited experience on GI collection and in particular image interpretation can be assisted in performing proper processing and making informed decisions with AI guidance. In this thesis, a DL-assisted WebGIS framework was developed to detect buildings, to delineate their rooftop boundaries, and to compare with existing building vectors to serve for change detection, and accordingly database updating. The input aerial and satellite images provided by the users can be processed by using pre-trained DL models for building detection. The detected rooftops can be vectorised in the proposed system, and a change detection component reveals the alterations between the existing and the detected vector data. Thus, the framework supports vector modification and drawing, and final products are stored in a spatial database management system (DBMS). The framework is adaptable and various DL methods can be integrated into the framework for different image segmentation problems. It is expected that with the availability of such systems, more users can support the geodata collection, updating and analysis processes, which are crucial for different applications such as environmental monitoring, spatial planning, digital twin creation for land management and simulations, etc.

Keywords: WebGIS, Citizen Science, Image Segmentation, Deep Learning, Geodatabase Updating, Change Detection

ÖZET

BİNA VERİTABANLARININ GÜNCELLENMESİ İÇİN DERİN ÖĞRENME DESTEKLİ WEBCBS ÇATISI GELİŞTİRİLMESİ

Recep CAN

Yüksek Lisans, Geomatik Mühendisliği Bölümü
Tez Danışmanı: Prof. Dr. Ali Özgün Ok
Haziran 2022, 72 sayfa

Coğrafi veri toplama yöntemlerinde çeşitlilik son yıllarda önemli ölçüde artmıştır. Bu gelişmeye paralel olarak coğrafi veri tabanlarının güncellenmesi için otomatik yaklaşımlara olan ihtiyaç da artmıştır. Makine öğrenimi (ML) yöntemlerinin kullanımına ek olarak, alanında profesyoneli olmayan kişilerin ve gönüllülerin katkıları bu hedefe ulaşmak için son derece gereklidir. Web ve mobil tabanlı Coğrafi Bilgi Sistemleri (CBS), uzmanlığı olsun olmasın farklı özgeçmişe ve becerilere sahip kişiler tarafından sağlanan coğrafi bilgilerin (CB) kullanımı için gerekli araçları ve yöntemleri sağlar. Gönüllü coğrafi bilgi (GCB) yaklaşımlarına ve Vatandaş Bilimi projelerine artan ilgi sayesinde, profesyonel olmayanların katılımcıların CBS ortamına katkılarından kademeli olarak yararlanılabilir. Öte yandan, yeni algoritmalar ve platformlar, gönüllüler tarafından toplanan verilerin doğruluğunu artırmak için yarı otomatik olarak CB çıkarımına yardımcı olabilir. Derin öğrenme (DÖ) algoritmaları gibi güncel yapay zeka (YZ) yöntemlerini WebCBS ara yüzlerine dahil ederek, yarı otomatik coğrafi bilgi çıkarma görevi gerçekleştirilebilir. Sonuç olarak, coğrafi bilgi toplama ve özellikle görüntü yorumlama konusunda sınırlı deneyime sahip gönüllülere, yapay zekâ rehberliği ile doğru işlemeyi gerçekleştirme ve bilinçli karar verme konusunda yardımcı olunması mümkündür. Bu tezde, binaları tespit etmek, çatı sınırlarını belirlemek ve mevcut bina vektörleriyle karşılaştırarak değişiklik tespiti ve veri tabanı güncellemesine olanak sağlaması için DÖ destekli bir WebCBS uygulaması geliştirilmiştir. Kullanıcılar tarafından sisteme yüklenen hava veya uydu görüntüleri, bina tespiti için önceden eğitilmiş DÖ modelleri kullanılarak işlenebilir. Tespit edilen çatılar sistemde vektör formata dönüştürülebilmekte ve değişiklik tespit bileşeni kullanılarak mevcut ve tespit edilen vektör verileri arasındaki değişiklikleri çıkarmak mümkün olmaktadır. Geliştirilen uygulama ile vektör düzenleme ve çizimler desteklenebilmekte ve nihai ürünler bir mekânsal veri tabanı yönetim sisteminde (VTYS) saklanabilmektedir. Geliştirilen uygulama uyarlanabilir özelliklere sahiptir ve farklı görüntü bölütleme problemleri için uygulamaya çeşitli DÖ yöntemlerinin entegre edilmesi mümkündür. Bu tür sistemlerin kullanılabilirliği ile çevresel izleme, mekânsal planlama, arazi yönetimi ve simülasyonlar için dijital ikiz oluşturma gibi farklı uygulamalarda CB toplama, güncelleme ve analiz süreçlerini daha fazla kullanıcının destekleyebilmesi beklenmektedir.

Anahtar Kelimeler: WebCBS, Sivil Bilim, Derin Öğrenme, Uzamsal Veri Tabanı Güncelleme, Değişiklik Tespiti

ACKNOWLEDGEMENTS

I would like to thank my supervisor, Prof. Dr. Ali Özgün Ok, who has provided me with his knowledge and support throughout my work.

I would like to thank my jury members Assoc.Prof. Dr. Fatih Nar, Assoc.Prof.Dr. Nusret Demir, Asst. Prof. Dr. Murat Durmaz, for their valuable suggestions and comments.

I would like to thank my fellow researchers, Mehmet Büyükdemircioğlu, İlyas Yalçın, Gizem Karakaş and Beste Tavus for always motivating me and making me feel that they are always by my side.

I would like to express my gratitude and respect to Prof. Dr. Candan Gökçeoğlu for his contributions to both my academic and personal development.

I would like to thank İrem Yıldız, who is always by my side and supports me endlessly, whose love I always felt, and who I know will always be by my side.

Finally, I express my sincere thanks and respect to Assoc. Prof. Dr. Sultan Kocaman Gökçeoğlu, whom I have learned and continue to learn a lot from both academically and personally. I am grateful to her for the vision she conveyed to us and all the support she provided.

Recep CAN
Haziran, 2022

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABBREVIATIONS

CitSci	Citizen Science
CNN	Convolutional Neural Networks
CRS	Coordinate Reference Systems
DBMS	Database Management System
DL	Deep Learning
GI	Geographic Information
GIS	Geographical Information Systems
ML	Machine Learning
NN	Neural Networks
SE-Net	Squeeze and Excitation Network
VGI	Volunteered Geographic Information
WebGIS	Web-based Geographical Information Systems

1. INTRODUCTION

In this section, the main motivation and the objectives of this thesis are explained. The main concepts elaborated in this study are discussed briefly, and the organization of the thesis chapters is presented.

1.1 Motivation

Advances in geospatial technology enable researchers to easily access high resolution remotely sensed data. Information extraction and interpretation of such data is a fundamental research topic for many researchers. The provision of timely, accurate, and up-to-date information on building footprints and rooftop boundaries is of high importance for government authorities and many industries such as defense, telecommunication, insurance, etc. Many applications, ranging from the evaluation of financial loss after disasters to real estate taxation, military mission simulations, infectious diseases (e.g., COVID-19 pandemic), urban planning, etc., require building footprint information or rooftop boundaries for proper evaluation.

Manual extraction of building rooftop boundaries is a time-consuming and labor-intensive task, especially for urban areas that contain large numbers of buildings. According to the United Nations (UN), by 2050, metropolitan areas will inhabit 68 percent of the world’s population (United Nations, 2019). This reveals the need for extensive planning in building related problems including establishing and updating the building databases in an accurate and efficient way. Considering this prospect, automatic or semi-automatic solutions for building detection and updating are immensely required.

Thanks to advances in Information and Communications Technology (ICT), the web and mobile-based geospatial solutions have become essential for geospatial data collection and interpretation. Considering the amount of the data in regional applications due to the high resolution requirements, contributions of people from any background have also become essential for achieving this task. Thanks to the development and widespread use of collaborative approaches in geospatial applications, the task of geodata collection and analysis benefits from the contributions of non-mapping professionals. Such efforts are mentioned under different terms in the literature, such as participatory geographic information system (GIS), volunteered geographic information (VGI), crowdsourced geographic information, citizen science, etc. (See et al., 2016). In recent years, volunteered participation in geospatial initiatives has increased and a well-known example can be given from the OpenStreetMap (OSM) Project (Open Street Map, 2022), which has approximately 8 million participants. This situation gives an important opportunity for the accomplishment of tasks that are traditionally performed by mapping professionals in the field or with the help of remotely sensed imagery. Thus, the detection of changes in the field, and the precise determination and measurement of object geometries can be carried out by non-professionals as well.

Instead of terrestrial measurements, interpretation of Earth Observation (EO) data, particularly the optical images acquired on different platforms such as satellite or airborne can be easier for non-professionals to extract the geodata. As an example, easily recognizable objects such as buildings or roads can be delineated precisely by volunteers from almost any background. However, limitations or difficulties still exist, especially in urban areas due to the volume of the data or unclear object boundaries caused by shadows, image quality, vegetation, etc. Several studies in the literature (e.g., Can et al. 2020, 2021; Yalcin et al. 2021; Li et al., 2022) have shown that such tasks can be supported by spatial analysis and artificial intelligence (AI) methods, which facilitate the mapping speed and help to increase the spatial data quality. Thus, as a spatial-data related sub-branch of AI, the GeoAI has the potential to provide great support for geodata collection and analysis by both geomatics professionals as well as non-professionals (Janowicz et al., 2020; Li, 2020).

Yet, proper tools and platforms that are accessible and easy-to-use for non-professionals are needed to be developed to exploit the full potential of volunteer contributions and GeoAI. Considering its potential and essentiality, a web-based Geographic Information System (WebGIS) framework utilizing open source technologies was proposed here to elaborate how deep learning (DL) methods can assist in vector database updating tasks for building rooftops. The proposed framework is scalable and can be adopted by government institutions, mapping agencies and VGI collection and analysis projects. As an example, the framework can be used by government institutions for detecting newly constructed buildings (without proper construction permissions) or those that are under construction. In addition, mapping agencies can integrate the framework into their systems to automatically monitor and reveal the need for changes in their vector databases if any significant differences are detected in current aerial images by the DL assistance. The VGI projects, such as the OSM, can integrate the framework into their system so that the volunteers can add, edit, and update current changes that were detected by the framework. Rather than observing the changes manually, the volunteers can utilize the DL assistance and modify the model prediction results in parts that require modifications. The proposed framework reduces the time spent on manual tasks for the map updating process and uses DL for guidance to support both experts and citizen scientists.

1.2 Concepts

Although the definitions of GIS vary depending on the perspective and various notations exist (Maguire, 1991), the detailed definition by Goodchild (2001) can be used here as; "*A geographic information system (GIS) is best defined as a software system designed to provide services related to geographic data. Such services include storage, analysis, transformation, maintenance, editing, visualization, modeling, and many more.*". It can also be stated that "*Geographic information systems are a special class of information systems that keep track not only of events (e.g. environmental disasters), activities (e.g. construction), and things (e.g. facilities, institutions, or natural resources), but also of where they happen or exist*" (Adam, 2013). The main components of GIS are in general considered as software, hardware, data, methods, and people. People include different kinds of contributors to the system, such as mapping professionals (e.g., geomatics engineers, surveyors, etc.), GIS administrators, planners, land owners, or any other end users, etc. The functional components of a GIS are presented in Figure 1.1 (Huisman and de By, 2009).

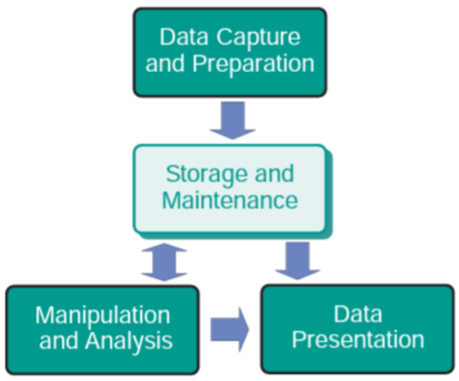

Figure 1.1Functional elements of Geographic Information System (Huisman and de By, 2009).

The VGI was first used by Goodchild (2007) to define the geographic information (GI) collected by users. The concept was also intertwined with neogeography, geographic citizen science, crowdsourced GI, mashup, participatory sensing, web mapping, etc. (See et al., 2016). The VGI projects have become increasingly popular with the developments in ICT technologies and the availability of low-cost positioning sensors on mobile devices, which have enabled nearly every location to become reachable. With the widespread use of mobile technologies, people from almost every country, language, and age group have become potential data providers. While VGI has been seen as a valuable approach in the literature for different applications such as land administration (Moreri et al., 2018), smart cities, digital twins, health, etc., concerns related to legal and ethical regulations, data quality (Antoniou et al., 2017), and reproducibility and replicability (Ostermann and Granell, 2017), exist.

Citizen Science refers to voluntary engagement in scientific processes, independent of technical or research expertise. Also, citizen science is used to describe groups or networks of individuals who serve as observers in a particular field of study (Goodchild, 2007). Scientists, applied users, and people working to achieve the Sustainable Development Goals of the 2030 Agenda are increasingly relying on data supplied by citizen scientists (de Sherbinin et al., 2021). Currently, a wide range of studies involve citizen science to cover data needs (Xu et al., 2022, Taylor et al., 2021, Senzaki et al., 2020, Yalcin et al., 2020). Considering the fact that several geospatial data collection efforts carried out by volunteers do not contribute to the scientific knowledge generation, i.e., inconsistent with the ten principles of citizen science compiled by the European Citizen Science Association (ECSA, 2015), it can be stated that the citizen science and VGI have common and even complementary aspects but are not completely overlapping (Kocaman and Gokceoglu, 2018).

GeoAI is a newly developing branch of AI, which benefits from the developments in AI techniques, hardware resources, high-performance computing (HPC) methods, and the availability of a large amount of high resolution geospatial data (Janowicz et al., 2020). In particular, the DL, computer vision and natural language processing (NLP) techniques support the progress (Janowicz et al., 2020). The three pillars of GeoAI defined by Li (2020) are shown in Figure 1.1.

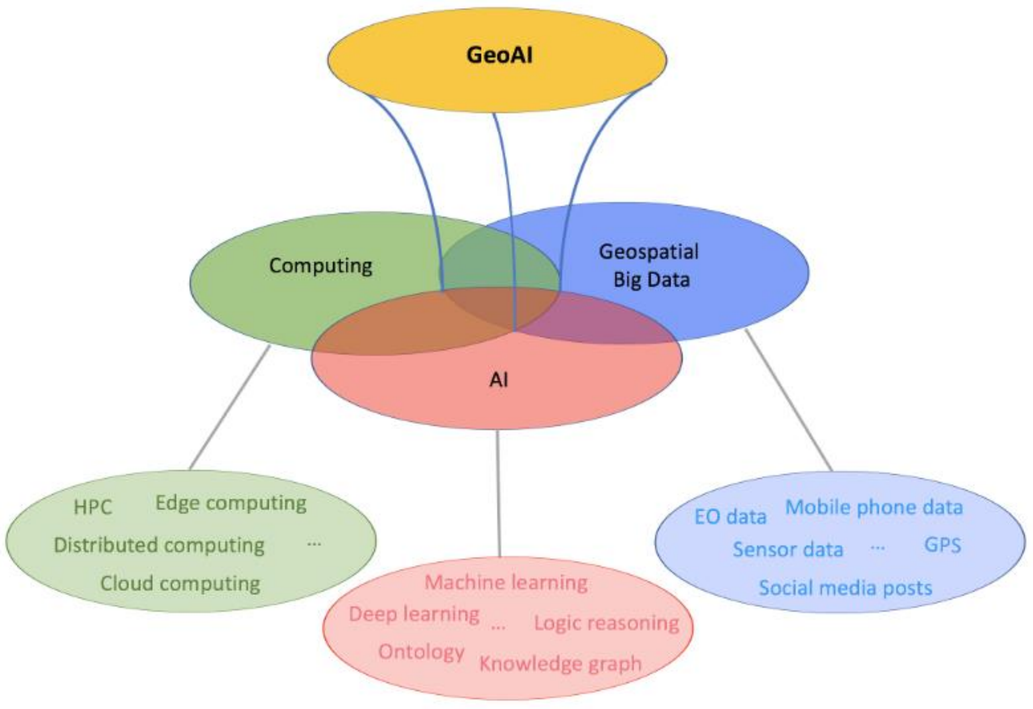

Figure 1.2. The three-pillar definition of GeoAI (modified after Li, 2020).

The DL is a machine learning (ML) technique that utilizes a neural network with three or more layers, and it is a kind of representation learning as illustrated in Figure 1.2. Representation learning is an approach which employs the ML to find not only the mapping from representation to output but also the representation itself (Goodfellow et al., 2016). The DL architectures involve multiple models in different layers, which allow learning from data at increasing degrees of abstraction (LeCun et al., 2015).

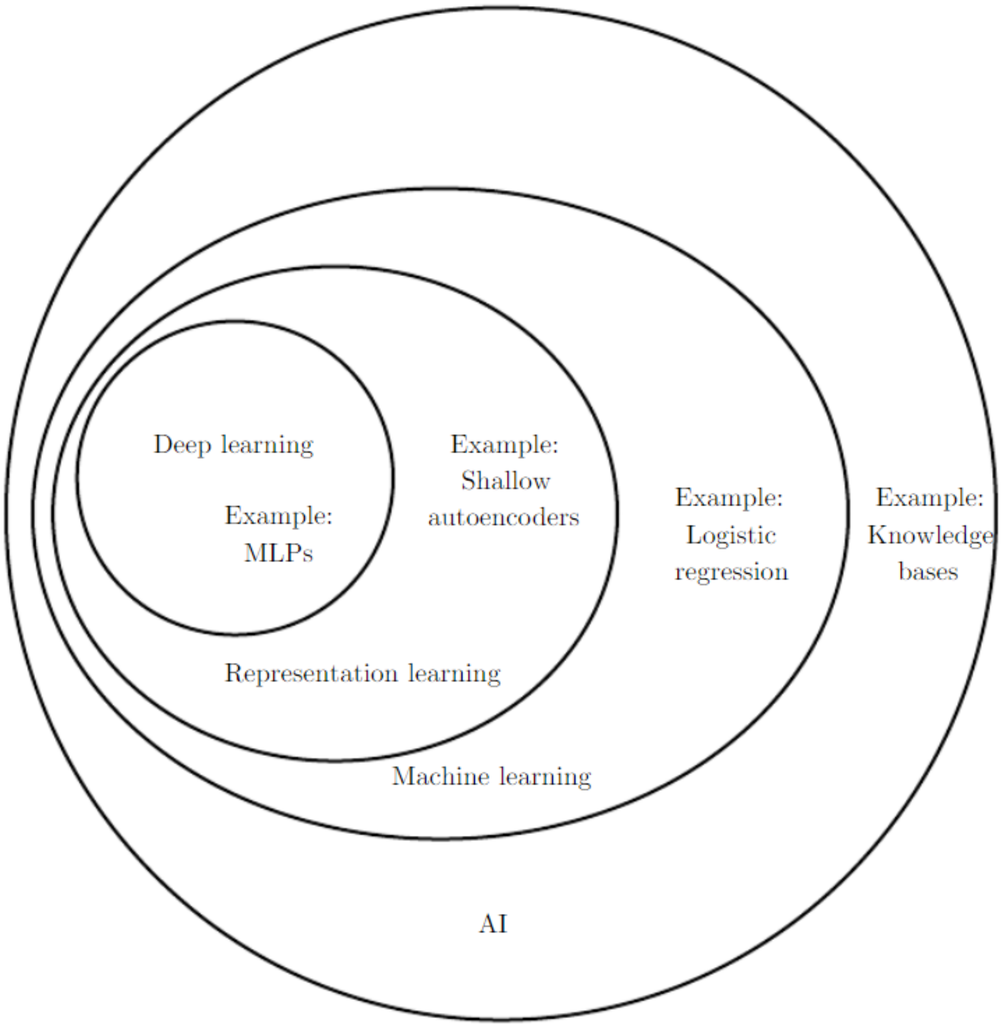

Figure 1.3. A Venn diagram that shows relationships in artificial intelligence (Goodfellow et al., 2016).

Figure 1.3 shows an illustration of a DL model. A computer sees an image as a collection of pixel intensity values. In order to identify an image, a function is needed to map such pixel values to object identity. The DL tackles this problem by decomposing the required complex mapping into a series of layered simple mappings, each of which is specified by a separate layer of the model (Goodfellow et al., 2016). The DL methods are making remarkable progress in overcoming problems that have long evaded the AI area. The DL architectures can successfully detect fine structures by modelling nonlinear relationships in highly dimensional problems and thus are useful for various application areas in science, industry and public institutions (LeCun et al., 2015).

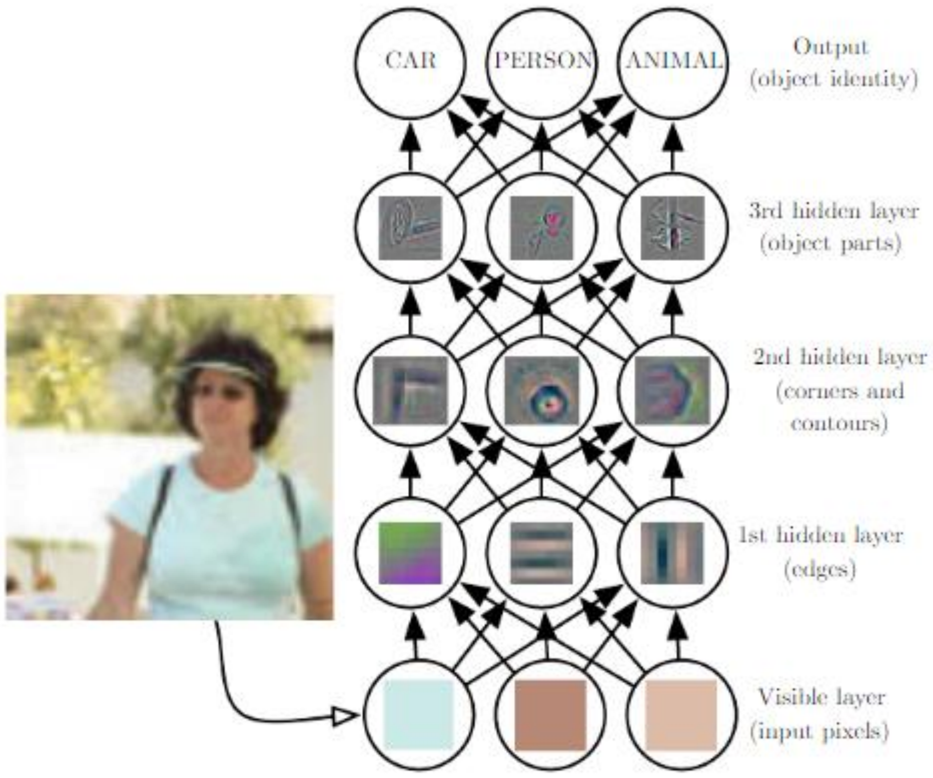
CAR
PERSON
ANIMAL
Output (object identity)
3rd hidden layer (object parts)
2nd hidden layer (corners and contours)
1st hidden layer (edges)
Visible layer (input pixels)

Figure 1.4. Deep learning model illustration (Goodfellow et al., 2016)

The Convolutional neural networks (CNNs) is a sub-class of neural networks, which models neighborhood information in a grid-like structure, and applicable to multitemporal datasets or imagery. The CNNs benefit from the local information by utilizing shared weights and pooling over a specified neighborhood in a DL structure with multiple layers (LeCun et al., 2015). Local connections enable every neuron to make connections between localized regions of the input grid instead of each cell of the grid. Shared weights are also known as kernels, which are defined weights of the corresponding feature map. Shared weights allow specification of similar features detected at various locations in the data. This approach may reduce the dimensionality, i.e. the unknown parameters existed in the network. Pooling simplifies the information that is provided from the convolutional layer (Nielsen, 2015).

A CNN is composed of several stages, as illustrated in Figure 1.4. For example, when considering dog identification problem using images, the first few layers of the typical convolutional network are responsible for extracting low level features like edges and corners. These features are then increasingly aggregated into more complex features that resemble the real things of interest, such as Samoyed dog or arctic fox in the last few layers.

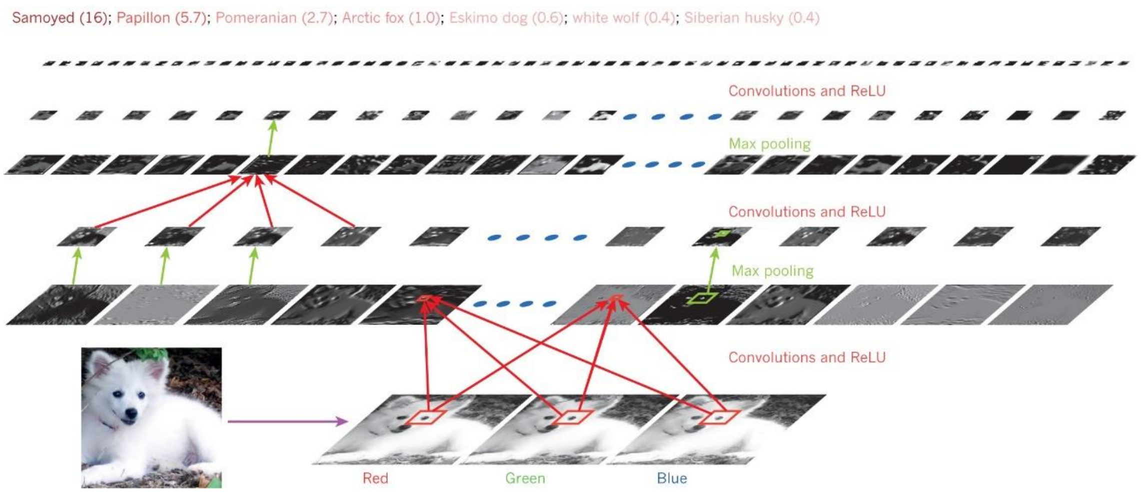
Samoyed (16); Papillon (5.7); Pomeranian (2.7); Arctic fox (1.0); Eskimo dog (0.6); white wolf (0.4); Siberian husky (0.4)
Convolutions and ReLU
Max pooling
Convolutions and ReLU
Max pooling
Convolutions and ReLU
Red
Green
Blue

Figure 1.5. Application of a CNN architecture to a Samoyed dog image (LeCun et al., 2015).

1.3 Objectives

The main objectives of this thesis are:

- to design and train a DL model for the automatic extraction of building rooftop boundaries from aerial imagery;
- to design a WebGIS framework and software architecture with the functionality of (i) vectorization of the detected building rooftops using existing methods in the literature; (ii) the detection of changes between two vector datasets; (iii) visualization of the aerial image and the vector data with the help of a web map viewer; (iv) vector data modification on the web map editor; and (v) data upload, modification and download;
- and to develop a system prototype with the required tools and user interfaces.

It is expected that the proposed system can demonstrate the potential of the GeoAI for building updating and unveil potential issues for further research and development.

1.4 Organization of the Thesis

This thesis includes six chapters. After the Introduction, the related works in the literature are presented in Chapter 2. Chapter 3 provides the technical background and the main components required in a WebGIS platform with a focus on GeoAI functionality. Chapter 4 presents the overall workflow and the details of the proposed WebGIS framework and the DL method employed for building extraction. Examples of the application interface are also given in this Chapter. Discussions are provided in Chapter 5. Conclusions and future work are given in Chapter 6.

2. RELATED WORK

In this section, the recent literature on the DL methods for building segmentation from aerial images together with WebGIS platforms developed for Citizen Science and VGI projects is provided.

2.1 The DL Architectures for Building Extraction

Deep convolutional neural networks (DCNNs) were often utilized for semantic segmentation and classification of aerial imagery (e.g., Shi and Zhu, 2018; Bittner et al., 2018; Wu et al., 2018; Yang et al., 2018). Several studies (Ronneberg et al., 2015; He et al., 2017; Chen et al., 2018) demonstrated remarkable performance in pixel based classification problems by using appropriate training datasets and the DL architecture. The outputs, on the other hand, are still rarely used or favored in an end-to-end system that aims to update a geodatabase. However, the DL-based methods have a high potential for semi-automated assistance to both experts and citizen scientists by recommending classification tags, verifying tags, updating notifications, quality analysis on the output, change detection caused by natural hazards, and so on. Allowing the usage of DL models to update a geodatabase would considerably increase the efficiency of human interpretation operations, especially for identifying areas of change. .Thus, time and labor expenses may be significantly reduced and interactive techniques can assure more precise and semantically complete data in a shorter period of time.

Marmanis et al. (2018) proposed a DCNN architecture that clearly specifies and extracts the boundaries between many semantic classes while segmenting high-resolution airborne images. A variety of semantic segmentation architectures were investigated in the study, including the usage of class boundaries, multi-scale processing and multi-network ensembles. The proposed model, whose example predictions are shown in Figure 2.1, had the best F1-score of 95.2 percent for the “Building” class in the ISPRS Vaihingen benchmark dataset.

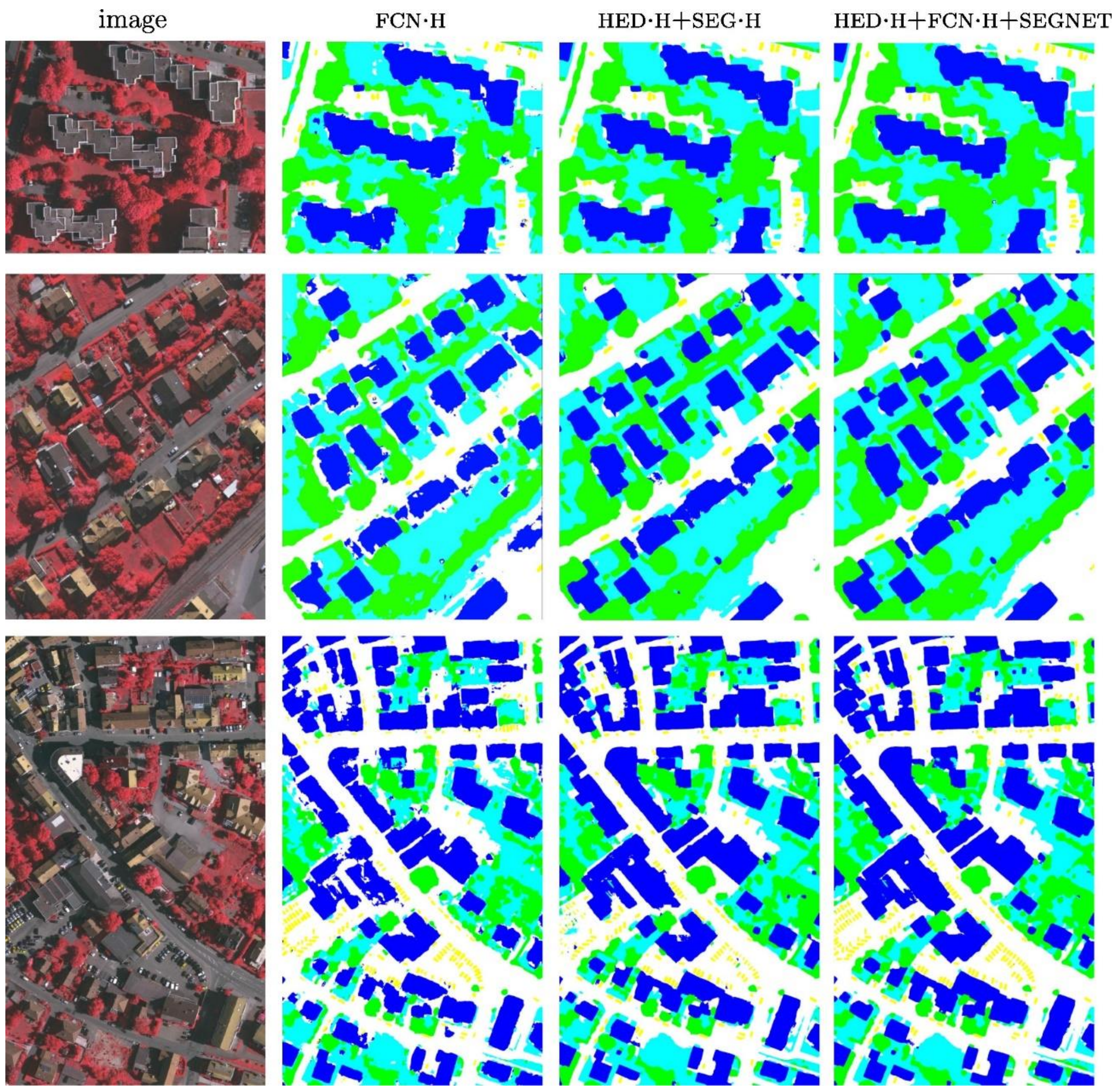

Figure 2.1. Example predictions in ISPRS Vaihingen benchmark dataset (Marmanis et al., 2018).

Yi et al. (2019) proposed a novel CNN architecture called DeepResU-Net that successfully performed pixel wise urban building classification from very high resolution imagery and produced accurate results. DeepResU-Net achieved higher F-measure, Cohen’s kappa score, and overall accuracy (OA) by 3.52, 4.67, and 1.72 percent respectively than U-Net architecture. Jiwani et al. (2021) used a modified version of the DeepLabV3+ architecture with a dilated ResNet backbone to propose a novel approach for building footprint extraction from satellite imagery. The proposed approach achieved state-of-the-art outcomes with 92.6, 96.3 and 83.4 percent F-measure respectively, on benchmark datasets, resulting in better quality graphics independent of the image resolution, scale, or urban density.

Li et al. (2021) incorporated multiple DL models to extract building footprint polygons from airborne imagery. The proposed method uses a pixel-by-pixel approach to compare model accuracy and generates building footprint geometries that are almost fit to the ground truth data as for edges, vertices, and shapes with 92.6,85.1 percent precision and confidence respectively in the WHU building dataset. Kada and Kuramin (2021) used PointNet++ and KPConv to classify building rooftops with several classes using point cloud data acquired by airborne laser scanning (ALS) with an Jaccard score of 0.948.

Another frequently utilized strategy for semantic segmentation problems using the DL is to combine several types of data sources and spectral bands. Through data stacking, different kinds of data sources can be merged with the RGB and provide additional information while training the DL architectures. Without changing the model structure, feeding the n-band input instead of three-band (RGB) input enables to use additional data. Hazirbas et al. (2017) proposed FuseNet, which is a prominent architecture for data fusion. The authors claimed that the depth information was not properly capitalized through stacking the RGB and depth information. Sun et al. (2021) used a frame field learning approach with the combination of true orthophotos and normalized digital surface models (nDSMs) to extract building outlines automatically. The height information provided by nDSM improved accuracy and regularity and achieved 70 percent average intersection over union score on the test set. Incorporating height information to RGB improved the IoU score by 12 percent when comparing the RGB images only results.

Bittner et al. (2018) were able to incorporate spectral (RGB and PAN images) and elevation information (nDSMs) from various sources and segment buildings in complex urban areas using fully convolutional networks (FCNs), resulting in building masks with the same raster resolution and 85.5 percent overall accuracy. It was demonstrated that combining spectral data with additional data and using the proposed approach resulted in slight improvements, particularly when predicting the building outlines. However, the minor improvements were a huge step forward because previously proposed approaches for building extraction algorithms were not particularly flexible and could not be easily generalized across different urban areas with various building types.

Zhao et al. (2021) proposed an approach for extracting building outlines by utilizing CNN architecture for extracting features and a recurrent neural network (RNN) for polygon vertices decoding to generate regularized building outlines. This was achieved by combining traditional feature engineering based methods into a single end-to-end DL architecture. The authors have made several modifications to PolyMapper's (Li et al., 2019) work, including the backbone, detection, and recurrence modules improvements. Xu et al. (2018) proposed a new approach for extracting urban districts from very high resolution airborne images that incorporated the DL and guided filtering. The segmentation accuracy of the ISPRS Potsdam and Vaihingen datasets was improved by 0.43 and 2.9 percent, respectively, using the proposed approach.

Buyukdemircioglu et al. (2022) have utilized nDSM data together with the true orthophotos in order to extract building footprints using DL. The authors have investigated the effects of nDSM on extracting building footprints and also the performances of the U-Net and LinkNet architectures with different backbones, and showed that combining nDSM data with the true orthophotos has increased F1-score while decreasing validation loss significantly. They achieved a Jaccard score of 0.926 on test dataset using LinkNet architecture with ResNet-50 backbone trained on the dataset includes nDSM. The results for the test area produced by their proposed models trained on the nDSM included dataset are shown in Figure 2.2.

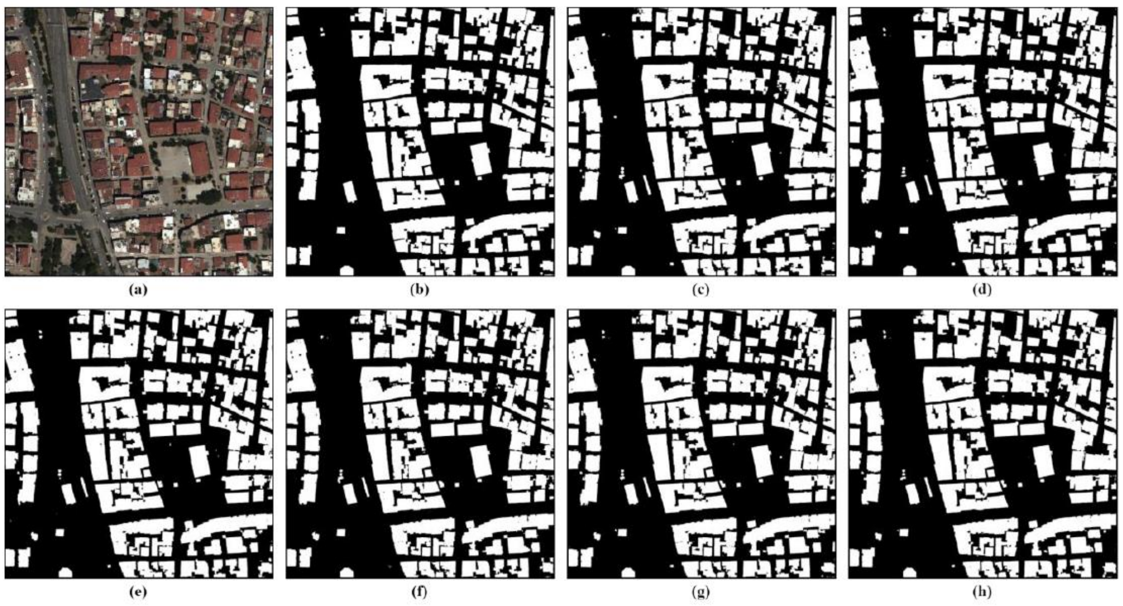

Figure 2.2. Results for the test area (a) True orthophoto, (b) Ground truth, (c-h) the proposed models(Buyukdemircioglu et al., 2022).

2.2 WebGIS Platforms for VGI and Citizen Science Applications

Although the DL-based techniques may greatly enhance intelligence for geodatabase updating task, the VGI obtained via mobile- and web-GIS platforms can also help DL architectures (Chen and Zipf, 2017). The DL methods and the WebGIS integration offer enormous promise for properly and quickly organizing, analyzing, and visualizing geospatial data when combined with the increased computing capacity of mobile devices. The WebGIS platform architectures are constantly developing as a result of advancements in computer technology and their requirements. Several review articles summarize the work done thus far (e.g. Agrawal and Gupta, 2017; Rowland et al., 2020) and interested readers may refer to those publications for further information. Decision-makers can also benefit from DL-based decision support systems including the interactive interfaces provided by such platforms.

Fan et al. (2021) proposed a platform that allows 3D building modelling using VGI data. A study conducted by Can et al. (2019) showed that the quality of the data provided by citizen science (CitSci) based data collection projects can be evaluated with the help of CNNs. In a recent work, Can et al. (2020) proposed a WebGIS framework that employs a CNN-based quality assessment tool in a GeoAI platform (called

GeoCitSci.com). In addition, Can et al. (2021) have proposed a framework for semi-automated geodatabase updating and demonstrated how DL based decision support systems would be useful for CitSci based applications.

Toro Herrera et al. (2021) have proposed a web platform that aims to enhance monitoring of water resources. The proposed platform allows uploading, publishing, and managing the water quality maps with the visualization of the water quality parameters map through the WebGIS interface. Balla et al. (2022) have also developed an analytical web tool that aims to determine and geovisualize water quality.

La Guardia et al. (2021) have proposed a system that aims to choose a new location for the P2G plant in the territory of Sicily in a semi-automated way. The system utilizes MATLAB and QGIS for GIS processing and employs the WebGIS tool for the visualization of a new P2G plant.

Atasoy et al. (2021) have proposed a 4D visualization and geo-analytics platform. The platform allows the visualization of multi-temporal and multi-platform geospatial data after disasters by providing user-friendly interfaces to geoscientists and citizen scientists. In addition to visualization, the platform also includes geo-analytic tools such as change detection for the assessment of post-disaster damage.

Nunes et al. (2022) have proposed a methodology that employs a WebGIS platform for the assessment of indoor radon potential. The WebGIS platform enables citizens to visualize indoor radon potential for a corresponding building location. In addition to visualization, the platform also includes a data collection option. The users are able to provide radon measurement data for the assessment of indoor radon concentrations.

Patera et al. (2022) have proposed a WebGIS application that aims to support marine spatial planning. The application allows users to visualize different kinds of conflicting activities that exist in the study area and employs five geoprocessing tools. Users can provide feedback about conflict assessment and management plans through the link involved in the WebGIS application to resolve conflicts in the study area. The authors have stated that the WebGIS application makes participation and active involvement of the public possible in marine spatial planning and that raises awareness.

3. TECHNOLOGICAL BACKGROUND

In this section, the technical concepts and technologies used in this thesis are briefly explained. Figure 3.1 shows the basic components of WebGIS in a client/server architecture. In the figure, the client defines the web browser or a mobile application. When a user accesses a geospatial application through a web browser, the respective client-side scripts and templates are requested from the web server. The web server sends the scripts and templates for the requested page to the client. The templates may include a variety of elements such as graphics, user interface, web map interface, or just plain text.

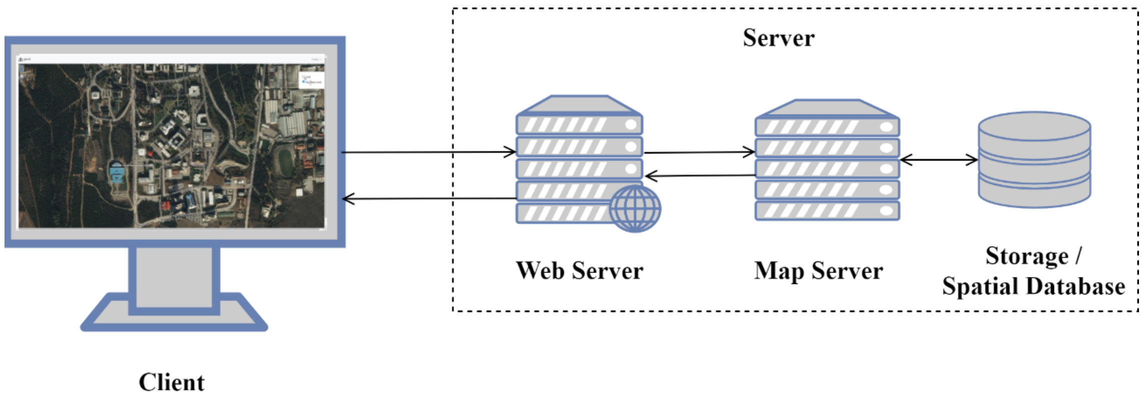

Figure 3.1. The basic components of a WebGIS.

The scripts provide the functionalities for the templates. For example, when a user clicks the *zoom in* symbol in the web interface, the respective function is triggered. The web browser runs the script and sends the request to the web server. The web server handles the request, and if the requested portion of the map is not present in the cache on the server, the data will be forwarded to the map server. The map server generates a response using the configuration parameters provided in the request and the data is parsed from the relevant data source, which can be a file storage or a spatial database. The data is then converted to the required map format by the map server prior to the delivery to web server. Finally, the script runs on the web browser on the user side to get the response from the web server and to finalize the zoom in task.

3.1. Client/Server Architecture

A client-server architecture consists of both a client and a server, with the client continually sending requests and the server responding to those requests (Oluwatosin, 2014). The requests and responses are handled via various protocols. Most commonly, the Hypertext Transfer Protocol (HTTP), which allows communication between client and server, is used. As a geospatial service, the Web Feature Service (WFS), which is an Open Geospatial Consortium (OGC) standard (OGC WFS, 2022) uses HTTP to create, modify, and exchange vector data on the Internet. As a data exchange format, the Geography Markup Language (GML) (OGC GML, 2022) enables the encoding and transfer of geographic information by WFS. Further data exchange formats and web services for geospatial applications can be found in the OGC resources (OGC Standards, 2022).

3.2. User Interface

A user interface enables the interaction of the user with the application. There exist numerous tools in order to design and develop user interfaces, such as Bootstrap (Bootstrap, 2021), Bulma (Bulma, 2022), Foundation (Foundation, 2022), etc. The Bootstrap is among the most popular open source front-end framework and includes several features that enable the developer to build and design responsive web pages quickly. Although several prebuilt components exist in Bootstrap, the requirement specific components need to be designed from scratch. In the proposed framework, JavaScript (JavaScript, 2022), HyperText Markup Language (HTML) and Cascading Style Sheets (CSS) were used to modify existing Bootstrap components and create new components from scratch.

3.3. Web Map Interface

Different from the generic concept of a user interface, a web map interface includes interactive web maps and respective functionalities. Currently, several JavaScript libraries such as Openlayers (Openlayers, 2022), Leaflet (Leaflet, 2022), Mapbox (Mapbox, 2022), Cesium (Cesium, 2022), and Google Maps (Google Maps, 2022) that allow to integrate 2D and 3D web maps into web applications are available. Openlayers is a very powerful library with a strong user and developer community. It is easy to use and has the ability to customize and extend its functionalities using 3rd party libraries such as OL-LayerSwitcher (ol-layerswitcher, 2022), jQuery (jQuery, 2022). The jQuery can be used for Document Object Model (DOM) manipulation and Asynchronous JavaScript and XML (AJAX) calls when requesting data from a web map server, and thus is preferred here.

3.4. Web Application Frameworks

The web application frameworks are composed of web services, web resources, and web application programming interfaces (API), and are designed to develop web applications. The Django (Django, 2022) and the Flask (Flask, 2022) are the most popular web application frameworks for the Python environment. Examples of the other frameworks that are currently available can be given as FastAPI (FastAPI, 2022), Tornado (Tornado, 2022), Sanic (Sanic, 2022), web2py (web2py, 2022), Masonite (Masonite, 2022).

The Django framework includes many features that perform common web development tasks such as authentication, content administration, etc. and thus makes web development easier. In addition, Django has a *contrib* module, which includes several frameworks for different tasks. One of them is GeoDjango (GeoDjango, 2022), which extends the Django model fields for OGC geometries and raster data, and utilizes functionalities for querying and manipulating geospatial data and vector/raster operations. Compared to Django, the Flask is more flexible and simple. The Flask requires fewer dependencies, so it is easy to develop small web applications. In addition, the Flask is also easy to extend for large scale web applications but requires further implementation while developing the application-based dependencies. For example, the Object Relational Mapping (ORM) framework is included in Django; thus, it is easy for the developers to handle the database operations easily. The Flask does not have the ORM framework and if a Flask web application needs a database operation, the developer must spend more time developing the database operations. Therefore, the Django technologies are preferred and utilized here.

3.5. Database Management Systems

Elmasri and Navathe (2015) defines database management system (DBMS) as "*A database management system (DBMS) is a computerized system that enables users to create and maintain a database.*". A DBMS usually has an interface that enables a user to perform various operations such as creating or deleting a database, querying, storing, updating data, etc. There exist several open-source DBMSs, namely PostgreSQL (PostgreSQL, 2022), MySQL (MySQL, 2022), MariaDB (MariaDB, 2022), MongoDB (MongoDB, 2022) etc. The PostgreSQL is one of the most popular open source DBMS with the PostGIS extension which enables spatial data storage and management (PostGIS, 2022). With PostGIS, it is possible to store geometries in the database and perform spatial queries. Several tools and features are offered through *psycopg* (pyscopg, 2022), which is a PostgreSQL adapter for Python. With the *psycopg* adapter, it is possible to use PostgreSQL features in a Python environment. Here, PostgreSQL and PostGIS were preferred for the implementation.

3.6. Web Servers

A web server aims to respond to client requests through transfer protocols such as HTTP, SMTP (Simple Mail Transfer Protocol), FTP (File Transfer Protocol). A web server is composed of software and hardware components. The software controls how a client accesses the hosted content, and the hardware stores the contents and allows them to be shared between connected clients. The Apache (Apache, 2022) and the NGINX (NGINX, 2022) are the most popular open-source web server software.

3.7. Web Map Servers

A web map server is software that allows communication between a web server and a database that includes geospatial data. A web map server aims to share geospatial data for web mapping applications. Examples of the most popular open source web map servers are the GeoServer (GeoServer, 2021) and the MapServer (MapServer, 2022). Unlike the MapServer, the GeoServer supports transaction operations in WFS, which is an OGC standard method to access and modify vector data over the web. A WFS transaction operation involves editing, so the operation enables clients to modify existing features or create new features in the corresponding data store.

3.8. Deep Learning Frameworks

A DL framework is composed of software packages that are used for designing and developing the DL models. The DL frameworks provide predefined layers such as convolutional layers, batch normalization layers, long short-term memory (LSTM) layers, activation layers, etc., and often include widely used DL architectures like VGG16, ResNet50 (e.g., see Tensorflow Keras Applications, 2022, Torchvision Models, 2022). In addition, most DL frameworks include extensions that can be used for many purposes other than the DL, such as signal processing, numeric calculations, image processing, etc. There exist many open source DL frameworks, such as TensorFlow (Abadi et al., 2015), Caffe (Jia et al., 2014), PyTorch (Paszke et al., 2019), and Apache MxNet (Chen et al., 2015). Although each of them has its own advantages and disadvantages, the TensorFlow was preferred here as it provides comprehensive documentation and has a strong user community. In addition, the TensorFlow provides a vast number of tutorials, samples, and codes. Thus, the DL implementation stage is rather easy and time-saving, which allows more time for the research and algorithm development by letting people to develop ideas faster in a short time period, which must be spent on how the framework works.

3.9. Python Tools for Geospatial Data Processing

The Python is a widely preferred programming language for geospatial applications. Many commercial and open source geospatial software provide Python APIs, such as ArcGIS from ESRI (ESRI, 2022), QGIS (QGIS, 2022), Geomatica from PCI Geomatics (PCI Geomatics, 2022), ENVI from L3Harris Geospatial (L3Harris Geospatial, 2022), ESA SNAP (ESA SNAP, 2022), Google Earth Engine (Google Earth Engine, 2022), etc. In addition to software, there are a large number of Python libraries that allow developers to implement their own software for geospatial processing. The GDAL/OGR (GDAL/OGR contributors, 2021), GeoPandas (Jordahl et al., 2020) and GeoAlchemy 2 (GeoAlchemy 2, 2022) are examples of the open source Python libraries that were also utilized in this thesis.

The GDAL/OGR provides data translation and processing utilities and supports a wide range of raster and vector data formats. With GDAL, one can georeference the DL predictions if the affine transformation parameters such as pixel resolution, upper left corner coordinates, rotations, etc., are known; or geolocate a satellite image using rational polynomial coefficients (RPCs). Also, GDAL can be used in vector operations. One can produce raster masks from GeoJSON or vectorize raster masks into vector formats or calculate intersections between two geometries. Many commercial tools, such as ERDAS from Hexagon (Hexagon, 2022), ArcGIS from ESRI, FME from Safe Software (Safe, 2022), MapInfo from Precisely (Precisely, 2022) employ the GDAL for different kinds of tasks (Software Using GDAL, 2022).

The GeoPandas extends the capabilities of the Pandas environment (the Pandas Development Team, 2020) in order to work with geographic data. The GeoPandas is used for processing vector data. In addition to the vector data processing capabilities, GeoPandas provides visualization tools. With the visualization tools, one can easily observe the results of the processing, visualize features that are included in the vector data, or make interactive maps. Since GeoPandas inherits many standard *pandas* methods, all functionalities of *pandas* work also in GeoPandas. The Geopandas includes many functionalities that are useful for geospatial analysis, such as overlay analysis, affine transformations, spatial joins, aggregations, and geocoding, etc.

GeoAlchemy 2 is a geospatial extension to SQLAlchemy (Bayer, 2012). The SQLAlchemy is a toolkit that allows developers to work with databases in Python. SQLAlchemy supports the object-relational mapper (ORM). The ORM allows developers to construct databases like modules following an object-oriented approach. The SQLAlchemy establishes connections between Python object models and databases. Basically, it maps the models into raw SQL queries and performs operations in the database. The GeoAlchemy 2 can be used for working with PostGIS. With the GeoAlchemy 2, PostGIS functionalities can be defined in Python object models like spatial queries, spatial relationships, etc.

4. PROPOSED WEBGIS FRAMEWORK

In this section, the user requirements and the overall workflow of the thesis implementation are explained, and the system architecture developed is presented in detail. In addition, the system components such as the web map interface, change detection, geospatial analysis, and the data management are described extensively. Furthermore, the DL and the vectorization methods employed here are provided, and several examples of the user interfaces are given.

4.1. User Requirements

Figure 4.1 denotes the use case diagram for the proposed DL supported WebGIS framework and defines the functional requirements for the framework. The data requirements vary depending on where the proposed framework is implemented. For simplicity, the user has been restricted to uploading the aerial image in GeoTIFF format and the vector data in ESRI Shapefiles (SHP) format. All uploaded data should include a Coordinate Reference Systems (CRS) definition. Additional raster and vector formats can be added to the system for extending the capability of working with a variety of data.

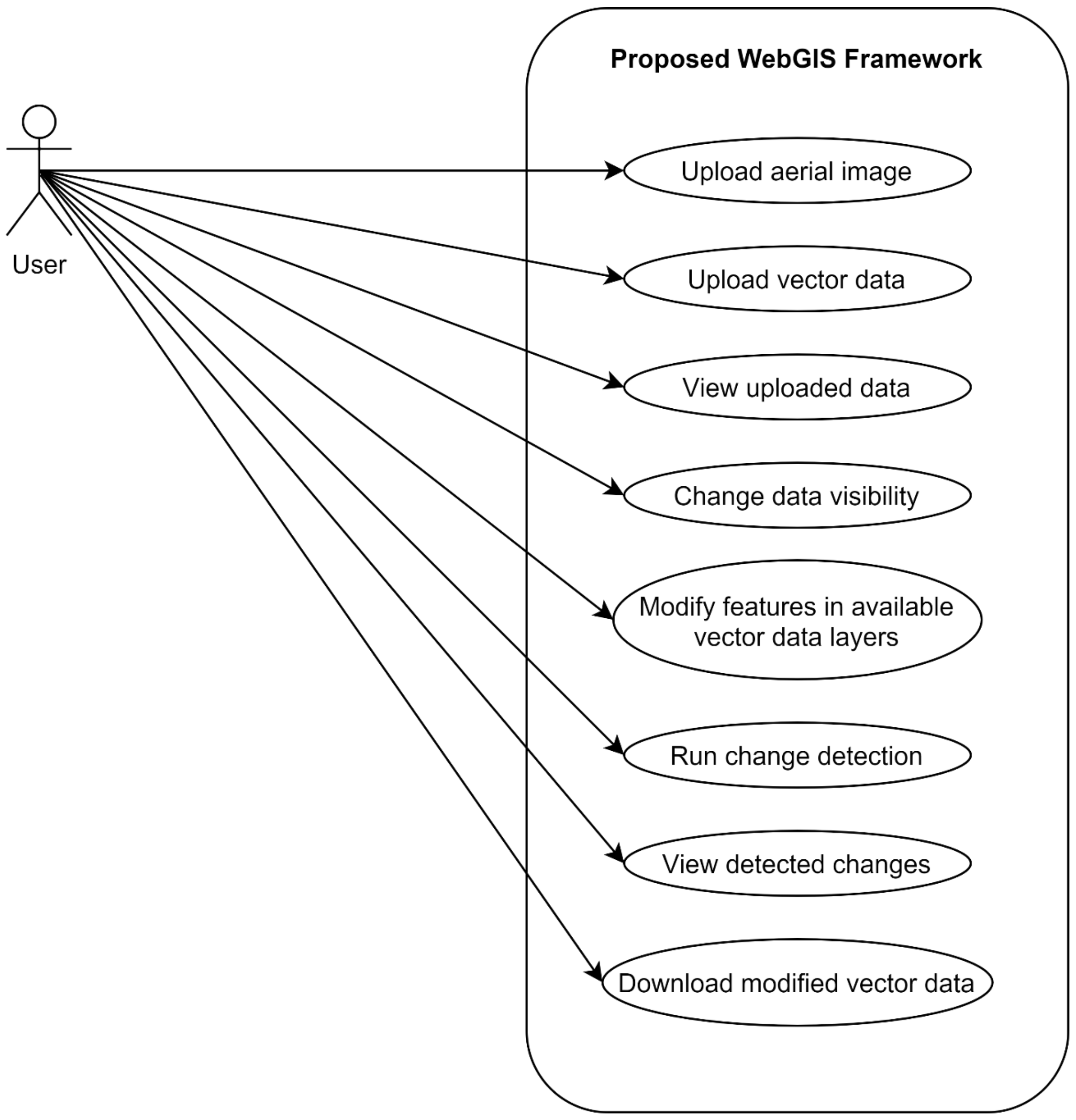

Figure 4.1. The use case diagram of the proposed WebGIS application.

The proposed system allows the user to view uploaded data and visualize the changes in existing vector data. The performance of the view functionality depends on the connection speed and the server capabilities, as well as the size and resolution of the uploaded data. A subsystem for modifying features in the vector layers is available in the proposed system. The questions of the number of concurrent users and the data security issues on the client side are left out in this thesis.

For the change detection functionality, the configuration of the server directly affects the performance of the framework. In the proposed framework, the user can run change detection, but the performance analysis is not assessed and left for further research and development. The following questions are left out:

- How many users can run change detection operations concurrently?
- How will server capacity be distributed when multiple change detection requests exist?

All the functional requirements listed in Figure 4.1 are satisfied by the proposed WebGIS framework. Since the aim of the proposed framework is to demonstrate the potential of GeoAI for updating buildings, a more generic requirements analysis has not been carried out and the performance of the framework was not assessed. The requirements will change depending on the number of users and the amount of data to be processed. The performance of the framework also depends on these requirements.

4.2. Overall Workflow

Considering the main user requirements explained in the previous section, the proposed WebGIS architecture is composed of four main components, namely change detection, geospatial analysis, data management, and a web map interface component (Figure 4.2).

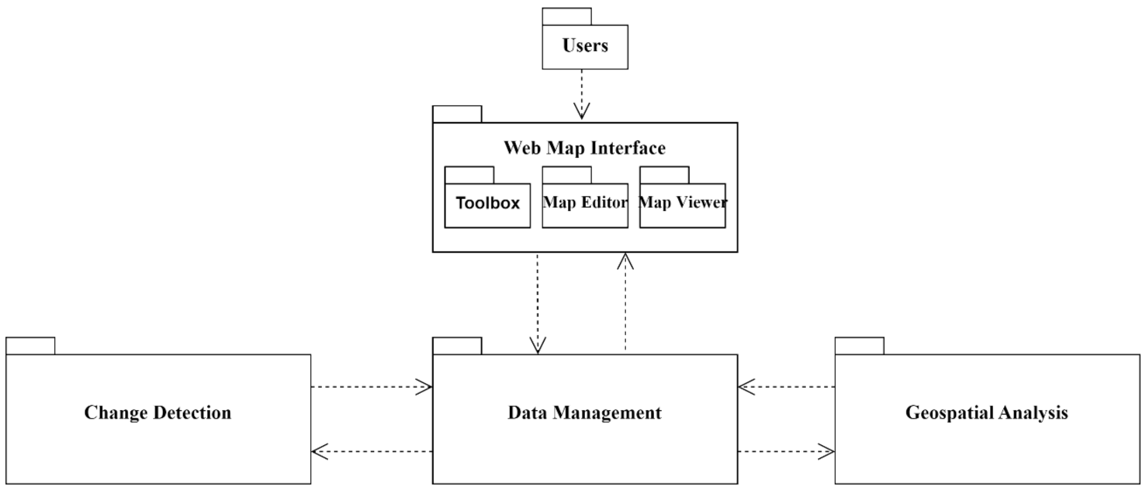

Figure 4.2. The system package diagram of the proposed WebGIS application.

The web map interface component includes three subcomponents, which are the *toolbox,* the *map editor* and the *map viewer*. The *toolbox* includes functionalities that allow the user to be able to upload data, run change detection and download updated vector data. The *map editor* enables users to modify features in vector layers that are available in the user session. The *map viewer* aims to show the active layer that is available in the current layer stack. The OSM and the Bing Maps Aerial layers (Bing Maps, 2022) are available in the proposed architecture. By default, the Bing Maps Aerial is shown to the user as base map and the user is able to change visibility of layers through the layer switcher in map viewer interface.

The data management component is responsible for controlling and managing the data. The component follows pre-defined validation rules before saving data to a database or file system. The validation step is only applied to the data that is provided by the user. The other components also use the data management component for data reading and writing purposes.

The change detection component requires a georeferenced aerial image and produces georeferenced vector data, which includes the building rooftop boundaries detected by the DL model. The tasks that are employed in the change detection component can be divided into preprocessing of aerial image, detection of building rooftop boundaries using the DL model and producing georeferenced vector data by vectorizing the raster outputs that are provided by the DL model.

The geospatial analysis component requires two sets of vector data. One of them is provided by the user and the other one is the output of the change detection component. The geospatial analysis component aims to find the differences between the two vector datasets and to create the georeferenced vector data from the differences. The output of the geospatial analysis component is the vector data that includes polygons. Each polygon is the difference between corresponding polygons in the two vector datasets.

The detailed description of each component is given in the next sub-section.

4.3.The WebGIS Architecture

The overall architecture of the proposed WebGIS application is given in Figure 4.3. A list of all technologies used in the framework is presented in Table 4.1. The application is composed of two parts, which are the client and the remote application server. The client part of the system provides an interface to the user. The user provides data, initiates tasks and requests data through the interface. All requests from the user are handled by the remote application server. Django, an open source web application framework, is used to enable communication between a client and a remote application server through HTTP.

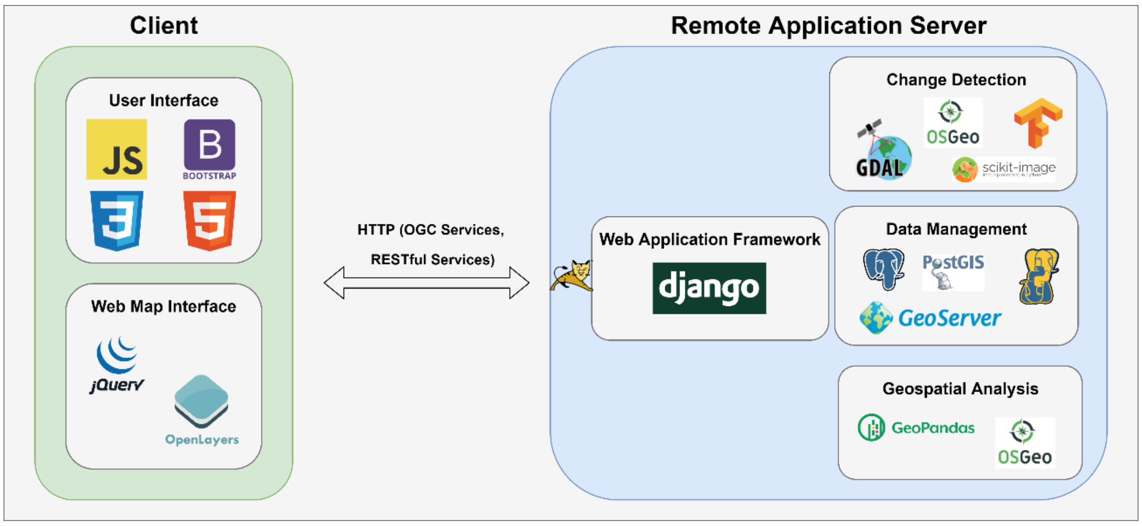
Client
User Interface
Web Map Interface
HTTP (OGC Services, RESTful Services)
Remote Application Server
Web Application Framework
Change Detection
Data Management
Geospatial Analysis

Figure 4.3. The overall system architecture of the WebGIS system with the technologies employed.

Table 4.1. A list of the technologies used in the proposed framework.

Technology	Main use purpose / functionality	Client/Server	Link
JavaScript	Providing functionality for UI components and Openlayers implementations	Client	https://www.javascript.com/
Bootstrap	Developing UI components	Client	https://getbootstrap.com/
CSS	Styling side bar	Client	https://www.w3.or

			g/Style/CSS/Overview.en.html
HTML	Structuring client side application	Client	https://html.spec.whatwg.org/multipage/
jQuery	DOM manipulation and AJAX calls,	Client	https://jquery.com/
Django	Running corresponding functions with handled request	Server	https://www.djangoproject.com/
GDAL/OGR	Reading raster data, georeferencing deep learning model predictions, vector data processing	Server	https://gdal.org/
scikit-image	Preprocessing and post-processing images	Server	https://scikit-image.org/
GeoPandas	Geometric operations for vector data and vector data processing	Server	https://geopandas.org/en/stable/
Tensorflow	Running pretrained model	Server	https://www.tensorflow.org/
PostgreSQL/ PostGIS	Storing raster and vector data	Server	https://www.postgresql.org/ - https://postgis.net/
GeoServer	Publishing vector and raster data	Server	https://geoserver.org/
GeoAlchemy 2	Using PostGIS geometry in SQLAlchemy	Server	https://geoalchemy-2.readthedocs.io/en/latest/

4.3.1. The Web Map Interface Component

The web map interface component runs on the client side of the system. The component includes user interface and web map interface elements. The user interface elements are composed of a navigation bar and a vertical side bar. Web map interface includes a map viewer element.

The navigation bar includes a modal element and a dropdown menu (Figure 4.4). The modal element, named “Detected Changes” in the interface, includes a list group that is formed by list items. Each list item is created dynamically after the change detection operation is finished. When the modal element is formed, it is shown to the user and added to the navigation bar. The user can click on each list item. The map viewer is triggered by this action and locates the user on the map where the difference occurred. In the initial stage of the system, the modal element is not available in the navigation bar.

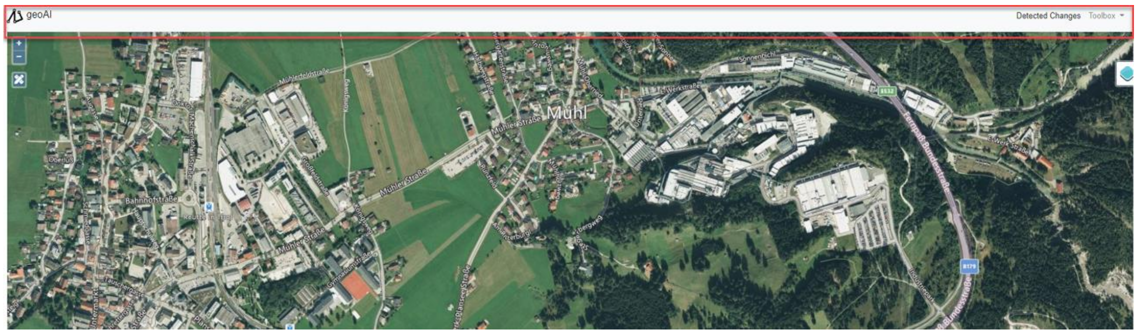

Figure 4.4. The navigation bar with the modal element and dropdown menu.

The dropdown menu, named “Toolbox” in the interface, includes items, which trigger modal dialog elements and functions. “Load Shapefile” and “Load Aerial Image” options under the Toolbox menu trigger the modal dialog elements (Figure 4.5). These elements include data description forms and enable the user to upload data. The other options, “Detect Changes” and “Download”, initiate change detection and download operations, respectively.

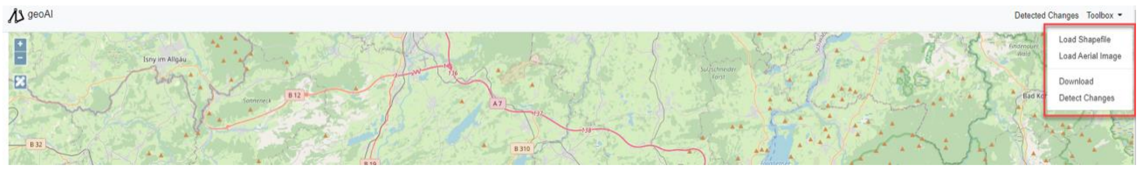

Figure 4.5. The dropdown menu (Toolbox).

The vertical side bar, named “Map Editor” in the interface, includes “Add”, “Edit” and “Delete” options (Figure 4.6). The options initiate related functions and enable the user to apply the selected option to available vector dataset. The vertical sidebar is only accessible if vector data exists in the current session.

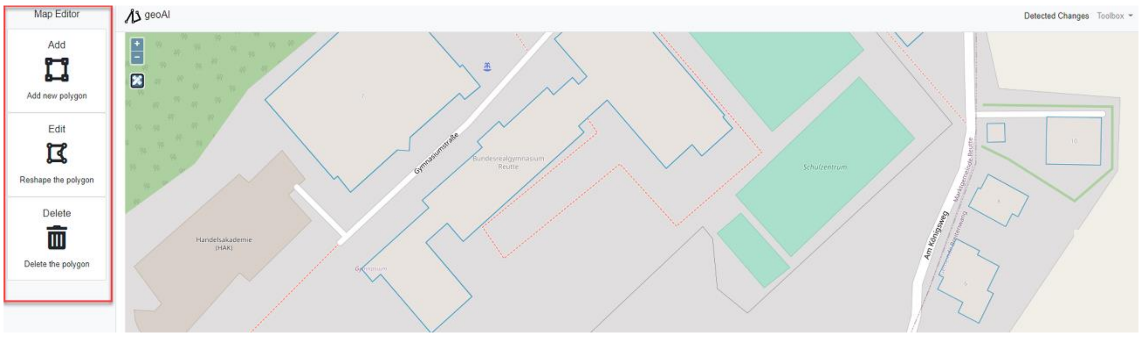

Figure 4.6. The Map Editor interface with the “Add”, “Edit” and “Delete” options.

The map edit button (Figure 4.6) opens the Map Editor menu and waits for the user’s selection. When the user clicks the map edit button, the icon of button changes. This means the selected operation of the map editor menu would be applied until the user clicks the map edit icon again. After clicking the button, all the modifications are applied to the existing vector data.

The map viewer element is composed of the viewer, zoom in-out buttons, a map edit button, and a layer switcher (Figure 4.7). The viewer is the interactive map area that is used for visualization and operational purposes. By default, Bing Map Aerial and OpenStreetMap are available as base maps. The user can visualize the uploaded data through viewer if the data was uploaded successfully to the system.

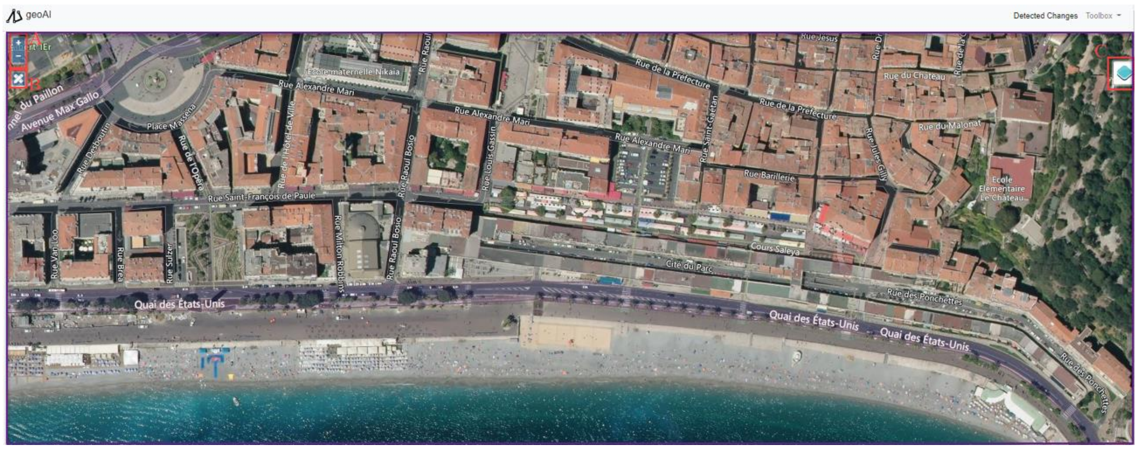

Figure 4.7. The map viewer element with the viewer (purple box), zoom in-out buttons (A), the map edit button (B) and the layer switcher (C).

The layer switcher (Figure 4.8) is a panel element that includes the list of existing layers in the system with checkboxes. Checkboxes enable the user to control layer visibility. From bottom to top, the order of the layers is as follows: “Bing Maps Aerial”, “OSM”, “Aerial Image (Uploaded)”, “Vector Data (Uploaded)” and “Results”. In the initial state of the system, the layer switcher only includes the “Bing Maps Aerial” and “OSM” layers. After each successful operation, the corresponding layer is added to the layer switcher and made available to the user. The user can activate layers through checkboxes. The viewer is triggered by the layer switcher and shows only active layers.

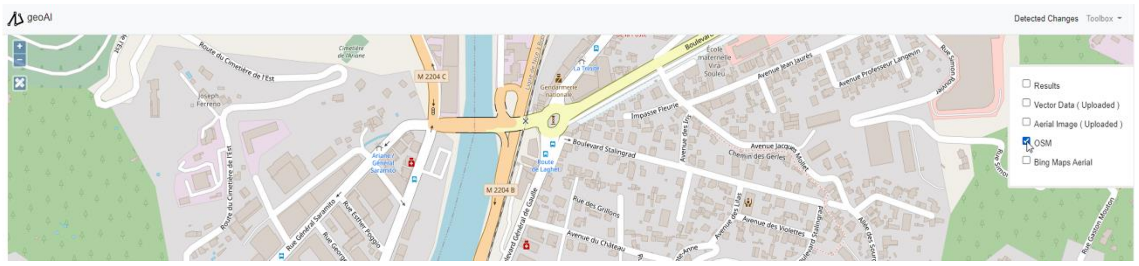

Figure 4.8. The layer switcher

4.3.2. The Change Detection Component

The change detection component runs on the remote application server. Figure 4.9 shows the workflow diagram of the change detection component. The initial task of the component is to request existing aerial image from the data management component.

After successfully obtaining the aerial image, the change detection component defines configurations for further processing steps. The configurations can be listed as; creating and validating temporarily available paths, creating a log file, validating pre-trained model weights file and setting predefined variables according to the image for use in the preprocessing step.

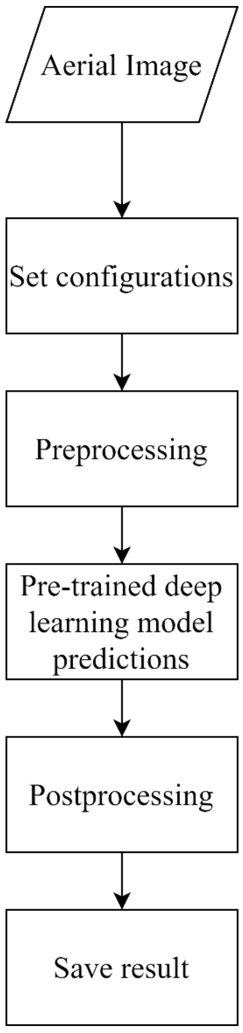

Figure 4.9. The workflow diagram of the change detection component.

After the completion of configuration setting, the component starts the preprocessing task. The preprocessing task is composed of scaling, tiling, and radiometric operations. The scaling aims to rescale the uploaded image resolution to the default image resolution. The default image resolution is equal to 30 cm, which is actually the resolution of the images that were used in DL model training. The tiling operation subdivides the uploaded aerial image by 256 x 256 so that each tile will be used in the pre-trained DL model. In order to reconstruct the uploaded aerial image, all steps and configurations such as tile index, overlapping ratio, etc. are saved to a temporarily available logging file. The final step of the preprocessing task is radiometric operations. In this operation, normalization and contrast enhancement are applied to each image tile.

All the image tiles that were produced in the preprocessing task are used in the pre-trained DL model. The pre-trained DL model produces predictions for each image tile. The DL architecture used in this thesis is explained in Section 4.4.

The post processing task includes Gaussian Smoothing, tile merging, georeferencing and raster to vector conversion. The Gaussian Smoothing is applied to each image tile prediction. After the completion of the Gaussian Smoothing operation, all image tile results are merged according to the configurations that were created in the preprocessing step. The output of the merging operation is the raster, which has the same extent and spatial resolution as the uploaded aerial image. Then, a georeferencing operation is applied to the resulting raster using the georeferencing information of the uploaded aerial image. Next, raster to vector conversion is applied to the georeferenced resulting raster by using the approach proposed by Sahu and Ohri (2019) with modifications that are explained in Section 4.5. The output of this operation is a georeferenced vector file that includes polygons created from the segmentation results. Finally, the vector file is saved to the database through the data management component.

4.3.3. The Geospatial Analysis Component

The geospatial analysis component runs on the remote application server. Figure 4.10 shows the workflow diagram of the geospatial analysis component. The geospatial analysis component is triggered when the change detection component finishes its processes. The geospatial analysis component requests two vector files from the data management component which are the user-provided vector file and the resulting vector file from the change detection component. When obtaining the two vector files successfully, the component starts the geometry validation process for each vector file.

The geometry validation process eliminates the possible irrelevant features such as lines, points, etc., and aims at selecting the polygon geometries from each dataset and creating a data frame for each selection. After the geometry validation, the component checks the CRS of each vector data frame to see if the two CRS are the same or not. If there are differences between two vector data frames' CRS, the component transforms one CRS to another so that the CRSs of the vector files will be the same.

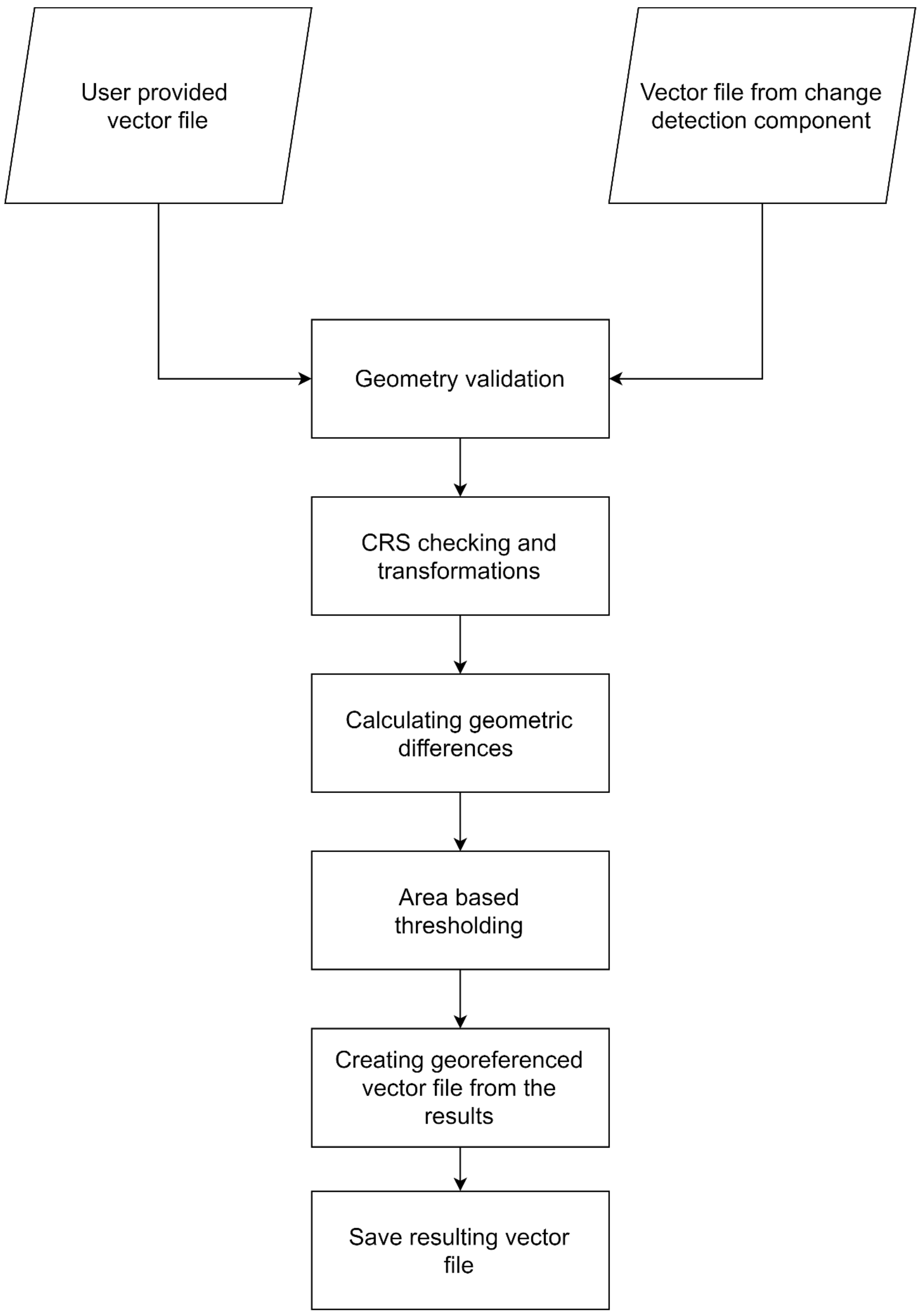
User provided vector file
Vector file from change detection component
Geometry validation
CRS checking and transformations
Calculating geometric differences
Area based thresholding
Creating georeferenced vector file from the results
Save resulting vector file

Figure 4.10. The workflow diagram of the geospatial analysis component.

After the CRS checking and transformation step, a spatial overlay is performed using the two vector data frames. The differences, which are the form of polygon geometry, are calculated for each feature in the vector data frame. After obtaining the difference polygons, area based thresholding is applied. Finally, a georeferenced vector file is produced using the polygons that were the outputs of the thresholding step. The resulting vector file is saved to the database through the data management component.

4.3.4. The Data Management Component

The data management component runs on the remote application server. The component aims to perform data related tasks, which are data input-output processes, data transactions, and management of the spatial database management system and the file system.

On the client side of the system, the component can be triggered by several operations such as loading a shapefile, loading an aerial image and modifying existing vector data. The "Load Shapefile" operation accepts a ZIP file that includes shapefile with the file extensions. When the zip file is uploaded through the "Load Shapefile" option, the data management component creates a temporarily available path according to the upload date. The component extracts ZIP file into the path and reads shapefile. After reading the shapefile, the component checks the geometries in the shapefile to determine whether the missing or invalid geometries exist or not. Next, the coordinate reference system is checked by the component. If the shapefile passes all validation steps, the component produces a well-known text representation of each geometry and saves the geometries to the PostgreSQL database. After successfully saving the shapefile, the shapefile is published automatically through Geoserver.

The "Load Aerial Image" operation triggers the data management component. In this operation, the component checks the aerial image and reads the geospatial information contained in the image. The component uses this information to define global variables such as spatial resolution, image width and height, CRS, etc. in the system so that the variables can be used by the other components. After validating the aerial image, the component creates a temporarily available path for the image and saves the image to that location. Next, the component saves the image path to the database and publishes the image automatically through Geoserver.

The other components also exploit the data management component for the corresponding data reading and writing processes. The data management component utilises PostgreSQL with the PostGIS extension as a spatial database management system for storing vector data and Geoserver for publishing both vector and raster data.

4.4.The DL Method for Segmentation

In this thesis, LinkNet based DL architecture was used. Instead of the original LinkNet encoder, a pre-trained squeeze and excitation network, SE-ResNeXt with a 32 x 4d template, was used as an encoder and integrated into the model. Figure 4.11 shows the DL architecture that was trained and used for segmentation.

The proposed DL architecture can be divided into three parts, namely the encoder, the skip connections, and the decoder part. The encoder part consists of the SE-ResNext-50 model, which was pretrained on the ImageNet database. The skip connections part includes four convolutional layers with a 1 x 1 kernel size. Each layer in skip connections part takes an input from the corresponding block in the encoder part and applies convolution operation. After the operation, the output of the operation is added to the corresponding block in the decoder part. The decoder part includes five blocks. Each block consists of convolution layers with a 3 x 3 kernel size and an up-sampling layer. Each decoder block takes inputs from the previous block and the corresponding layer in the skip connections part. After any convolution layers in both the encoder and decoder parts, batch normalization is applied to the outputs of the convolution layers.

The Inria Aerial Image Labeling dataset (Maggiori et al., 2017) was used for the model training. The training set includes color image tiles. Each tile has a dimension of 5000 x

5000 pixels with a 30 cm resolution. There are 180 image tiles. The image tiles were split into training (144 tiles), validation (16 tiles) and test (20 tiles) sets.

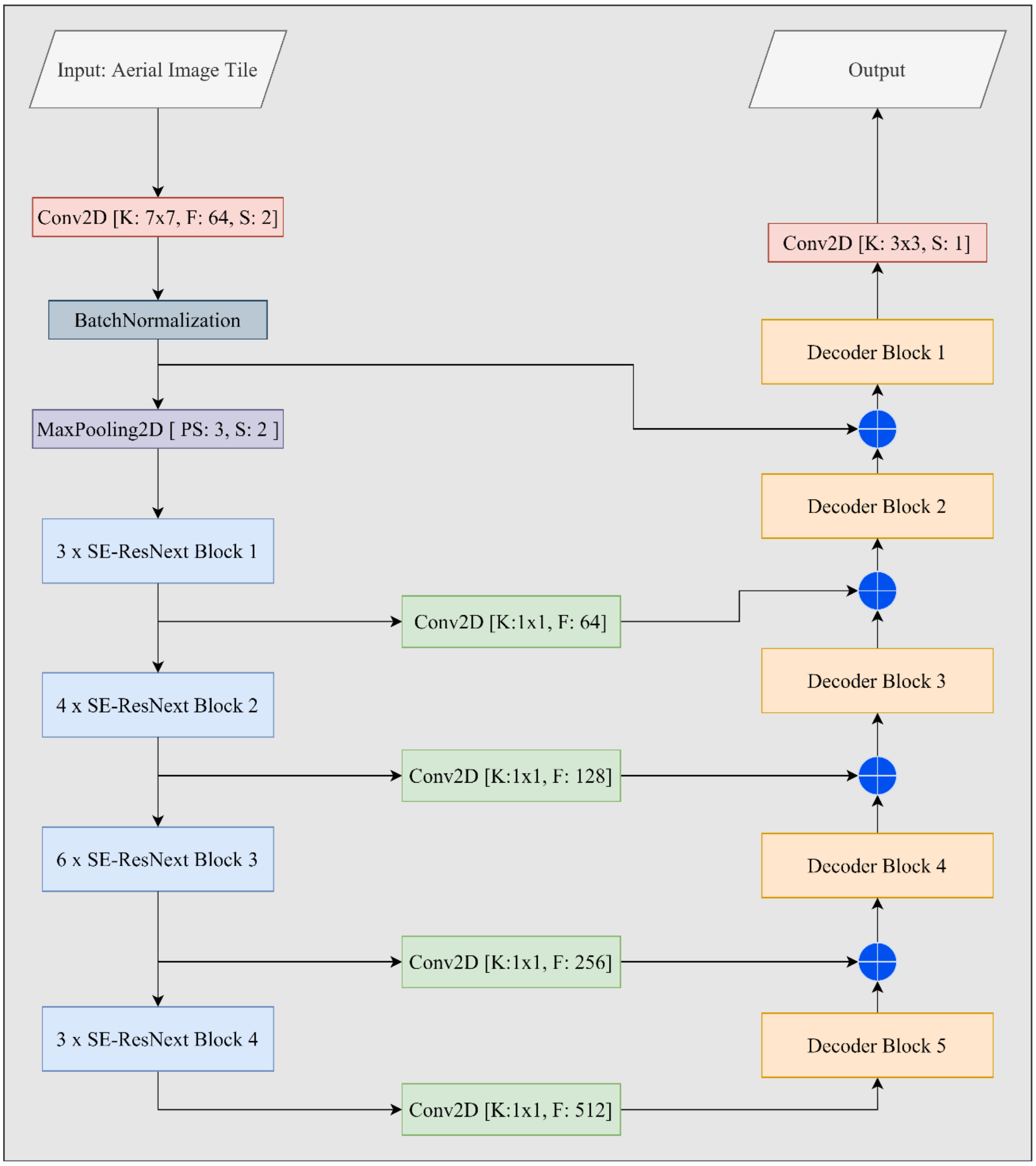

Figure 4.11. The DL architecture used for segmentation. K: Kernel Size (width x height), F: Filter Size, S: Stride, PS: Pooling Size

The images were cropped so that each image would have a size of 256 x 256. The radiometric data augmentation techniques, which are random brightness and contrast changing, gamma stretching, embossing, blurring, contrast limited adaptive histogram equalization, randomly dropping color channels, and multiplying image to random number, were applied to the batches of cropped images during the training process.

Because of computational resource limitations, the batch size was restricted to 8. The DL model was trained for 68 epochs using the RMSProp optimizer with cosine learning rate decay and the combination of binary cross-entropy and Dice loss.

4.5. Vectorization of Building Segments

An approach developed by Sahu and Ohri (2019) was followed here for the vectorization of building segments with some modifications. The building segments were vectorized using the GDAL/OGR library (GDAL/OGR contributors, 2021). For simplifying the building segments produced by the GDAL/OGR library, the Douglas-Peucker (Visvalingam and Whyatt, 1990) line simplification algorithm was applied. An iterative method was carried out in order to define a tolerance value that is the distance between the initial and the output geometries for the Douglas-Peucker algorithm,. The iterative approach seeks the tolerance value that minimizes the average area changes of the geometries in the building segments. After the simplification, the final building segments are generated and a vector dataset is produced. Not all the operations that exist in the Sahu and Ohri (2019) approach, such as the minimum bounding box estimation, separating the connected building vectors, etc., were needed in the raster to vector conversion in this thesis. In addition, the iterative search was not included in the Sahu and Ohri (2019) approach.

It must be noted that different approaches exist in the literature for the vectorization of building segments. Further improvements can be carried out as future research.

4.6. The User Interfaces

User interfaces of the implemented GeoAI platform are demonstrated in Figures 4.12 - 4.25. In Figure 4.12, the menu that enables users to upload vector and raster data is shown. In Figure 4.13, the layer selection menu is shown. By default, the vector data layer from the OSM and the Bing Maps Aerial layer are available in the platform. Figure 4.14 shows the demo vector data that were fetched from the OSM and uploaded to the platform. The user can change the visibility of the layer through the layer selection menu. In Figure 4.15, the map editor menu that allows users to add, modify, and delete features in available vector data is shown. The user can access this menu by clicking the map edit button if the vector data are available on the platform.

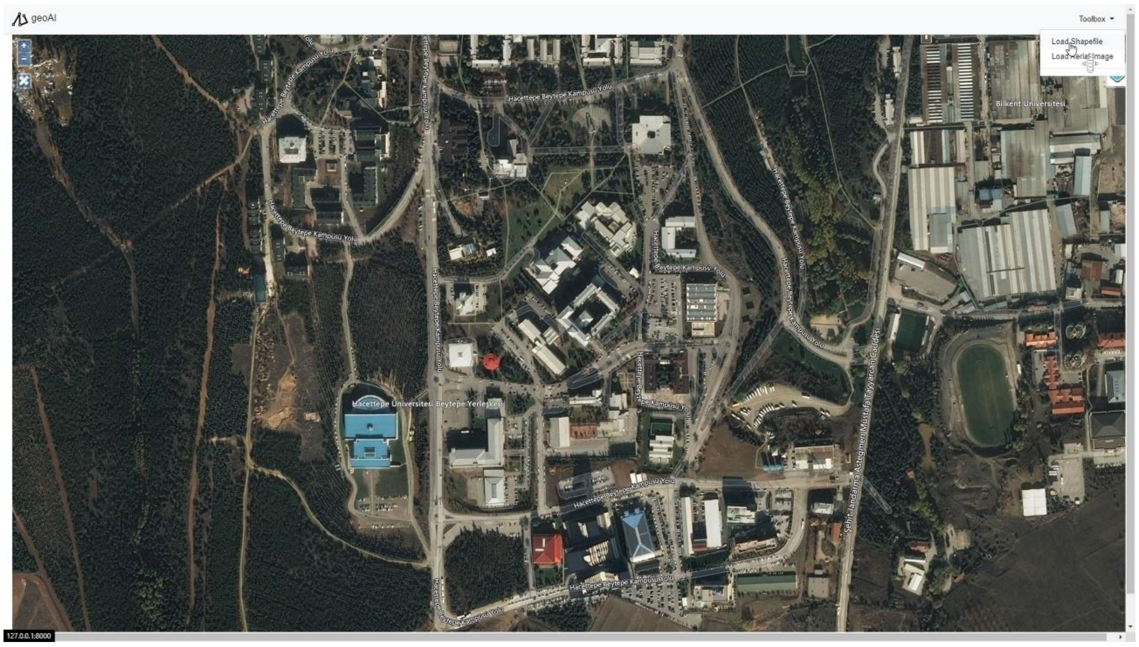

Figure 4.12. The menu for uploading vector and raster data (Can et al, 2021).

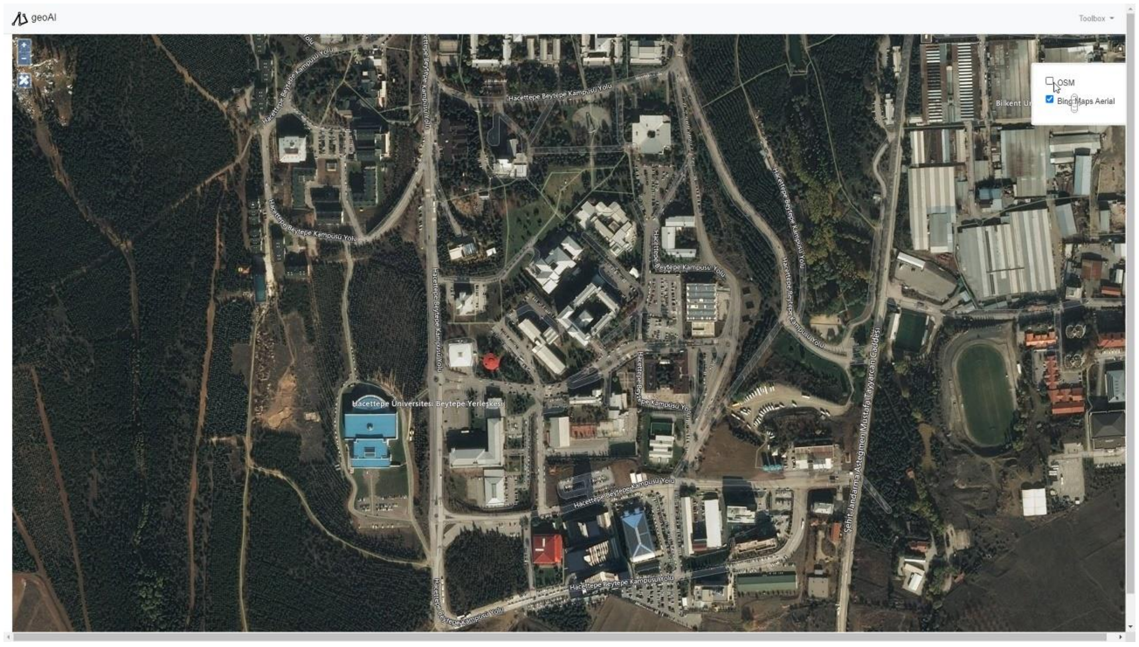

Figure 4.13. The layer selection menu (Can et al., 2021).

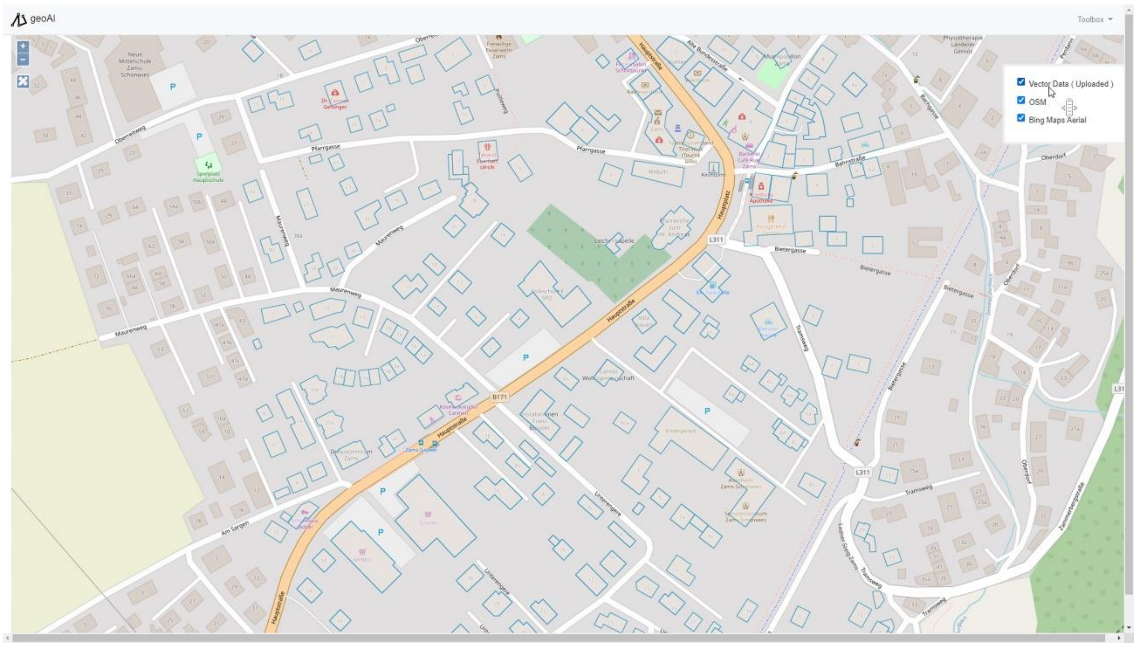

Figure 4.14. Uploaded vector data fetched from the OSM (Can et al., 2021).

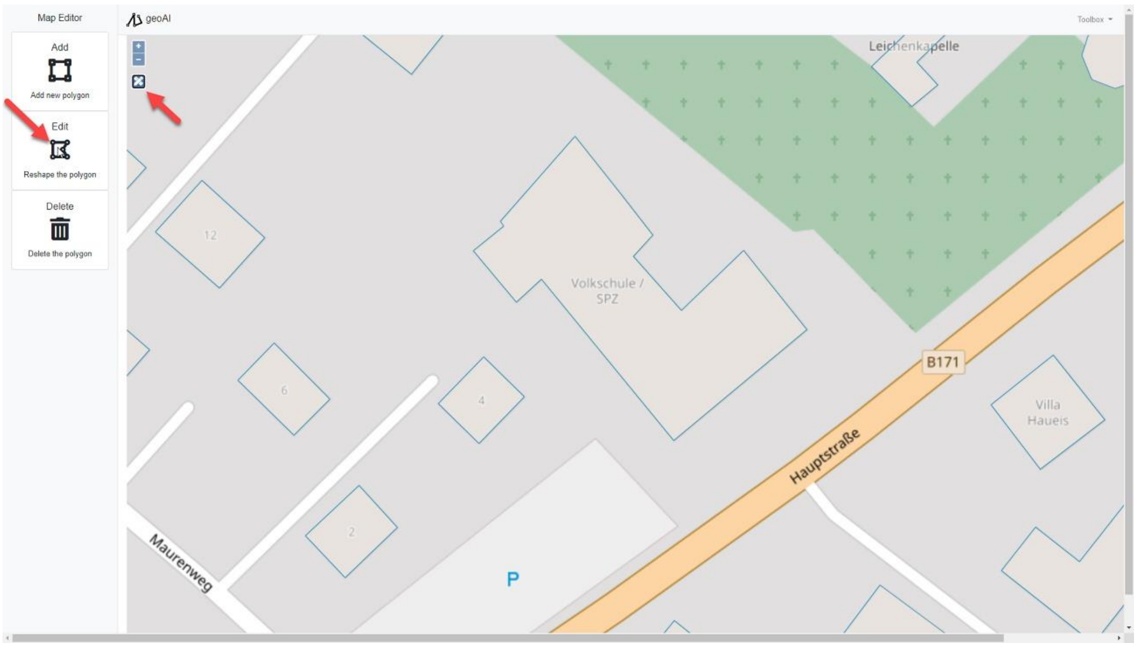

Figure 4.15. Map editor menu (Can et al., 2021).

In Figures 4.16 and 4.17, the modification operation on an existing feature through the map interface is demonstrated. Figure 4.18 shows the example raster data uploaded to the platform. In Figure 4.19, the “Detect Change” button is shown. The user can run a change detection operation by clicking this button. After finishing the change detection operation, a differences list that includes list items for navigating the user to the area of change is shown in Figure 4.20. The list can also be accessed by the “Detected

Changes” button on the navigation bar. The user can also observe changes by changing layer visibility through the layer selection menu as demonstrated in Figure 4.21. Figure 4.22 shows the missing building detected by the platform through a demo application. (Can et al., 2021). The user can use the map editor menu and add missing buildings by either drawing them from scratch or modifying the output of the change detection operation (Figures 4.22 – 4.24). In Figure 4.25, the download option for the updated vector data is shown under the toolbox menu.

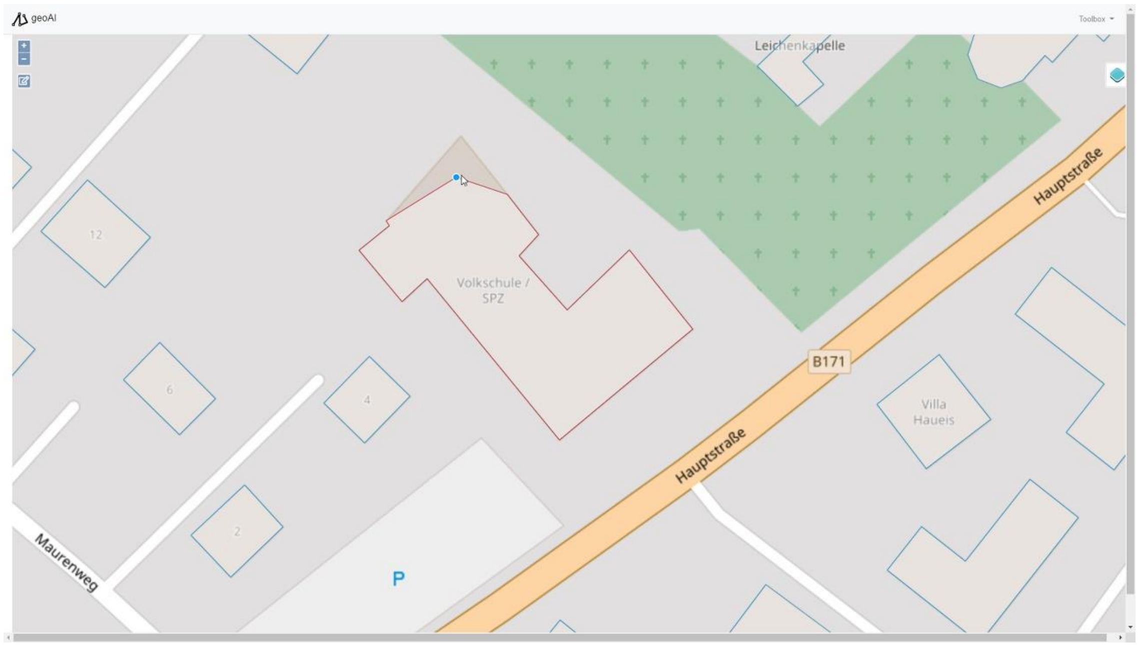

Figure 4.16. Modifying feature through web map interface (Can et al., 2021).

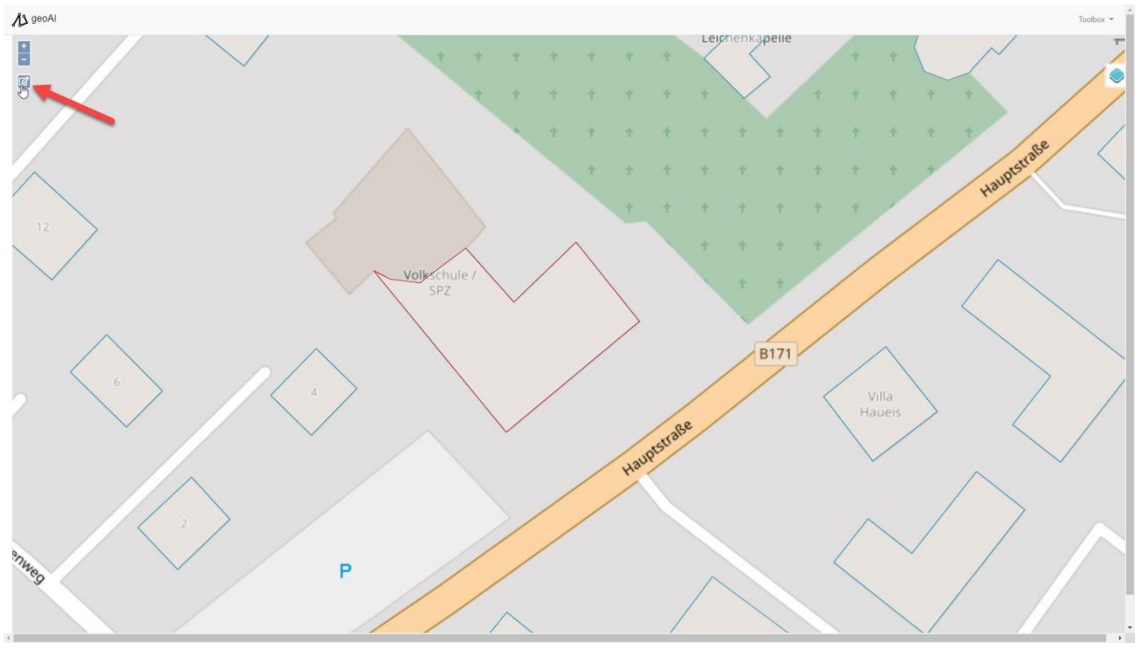

Figure 4.17. Finalizing the modification (Can et al., 2021).

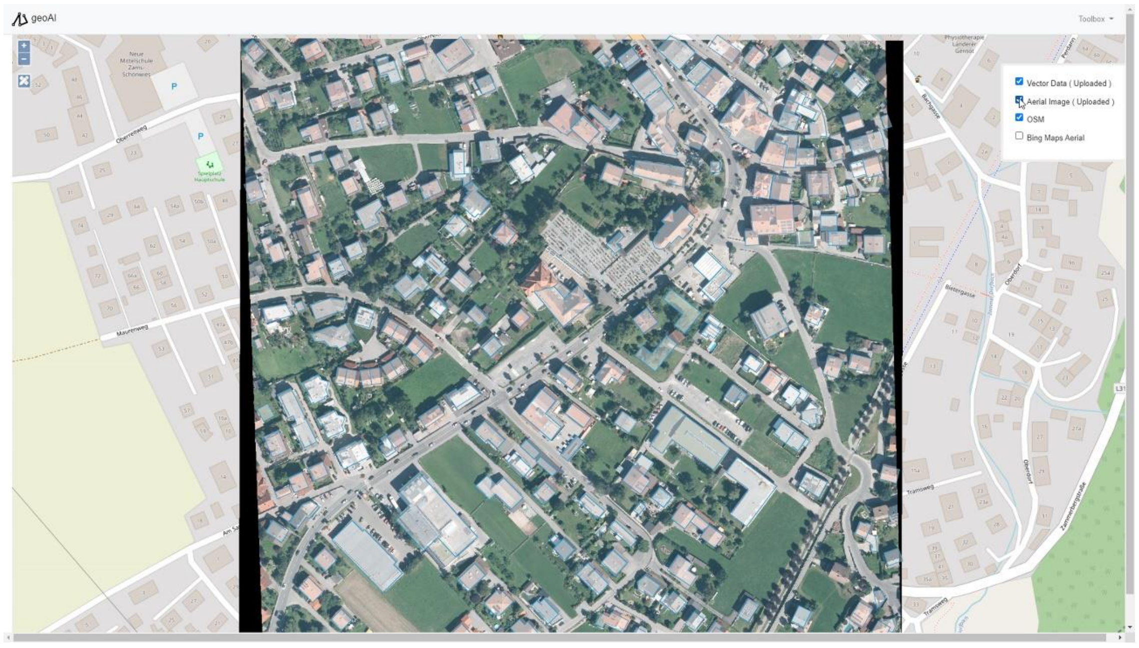

Figure 4.18. Uploading the raster data to the platform (Can et al., 2021).

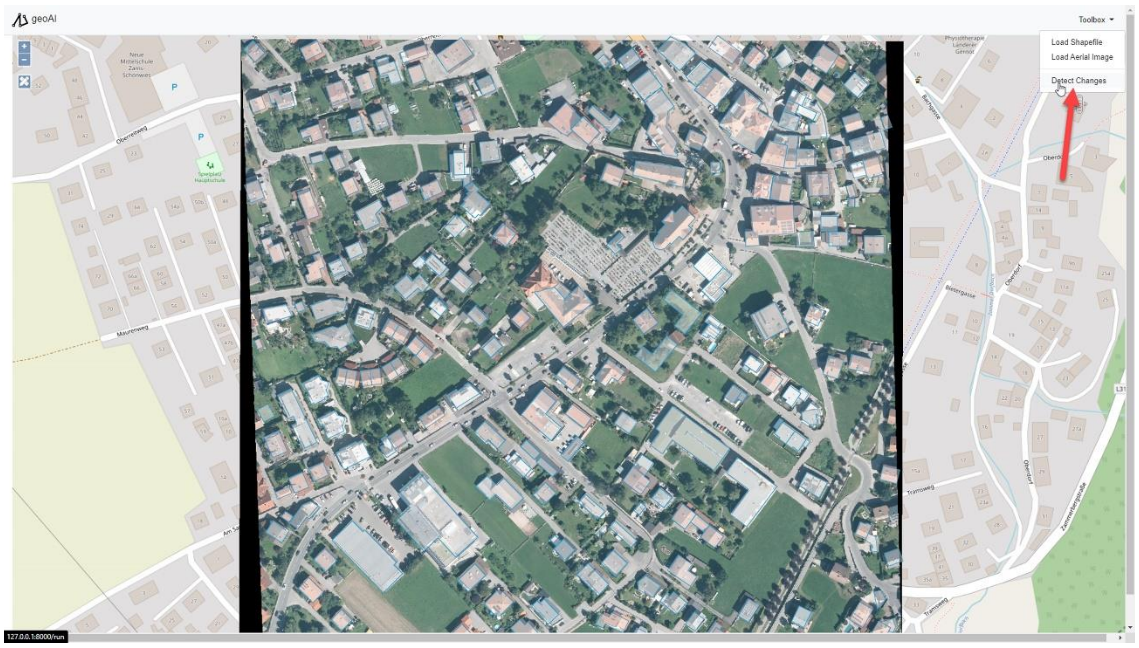

Figure 4.19. Change detection operation available in the toolbox menu (Can et al., 2021).

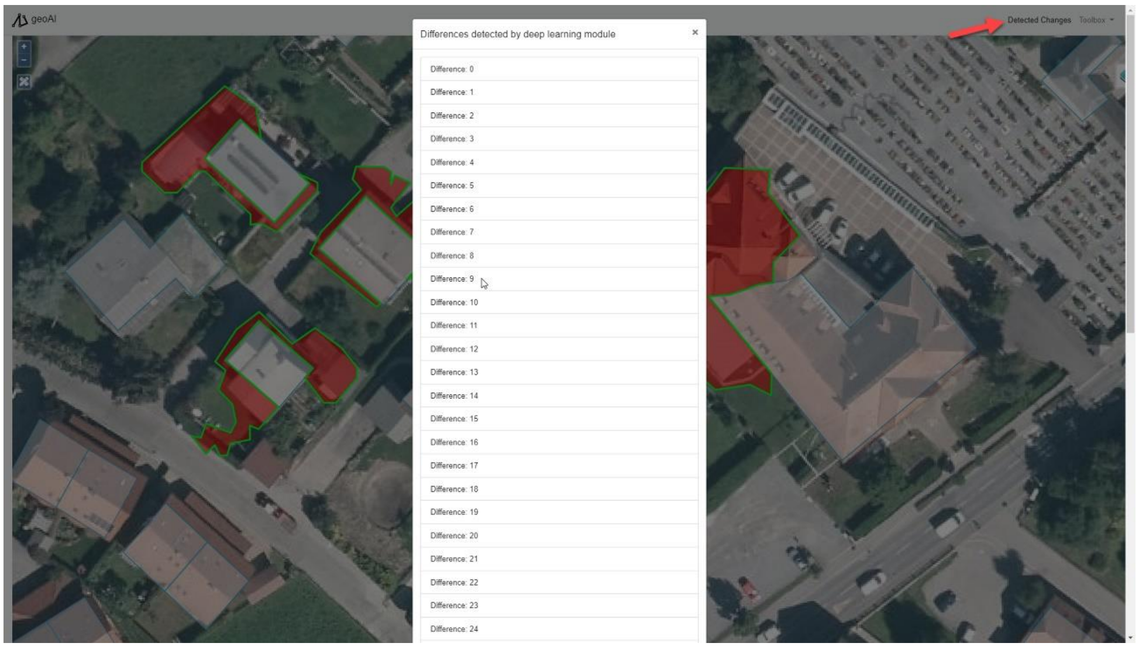

Figure 4.20. Differences list that shows the detected changes (Can et al., 2021).

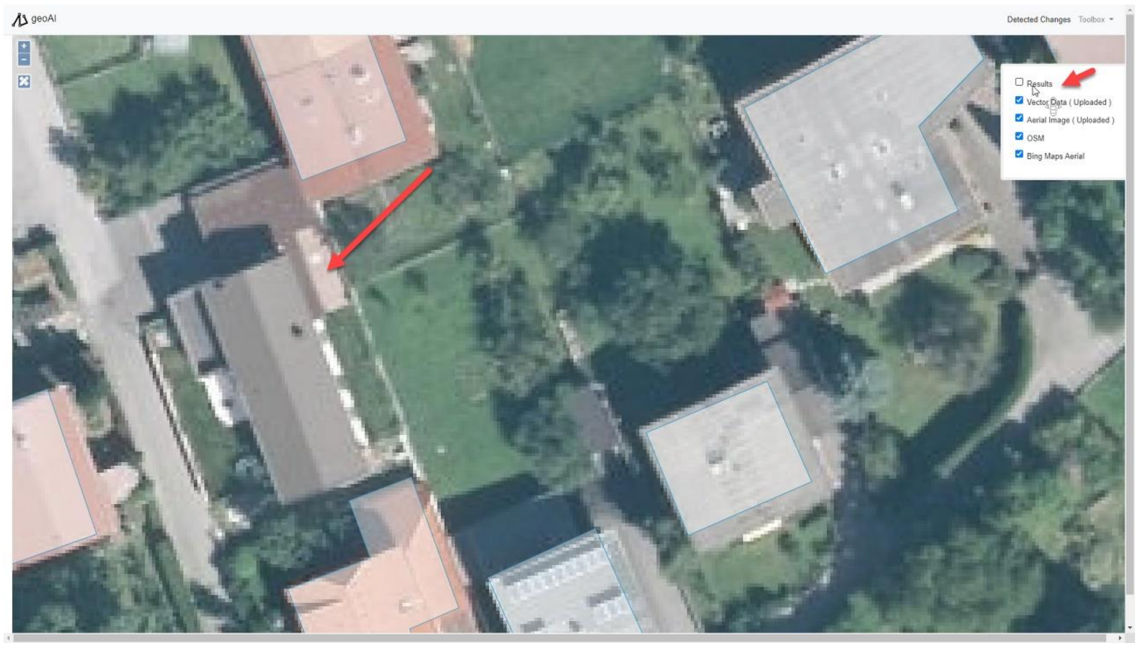

Figure 4.21. The output of the change detection operation is available as “Results” layer (Can et al., 2021).

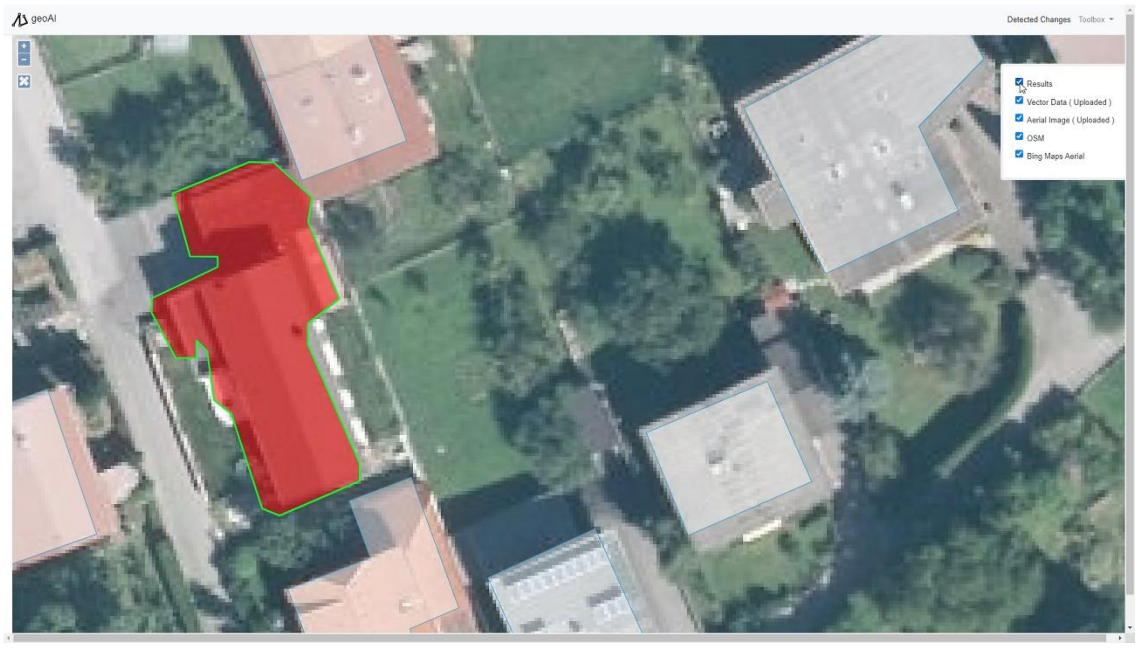

Figure 4.22. Missing building detected by the platform. (Can et al., 2021).

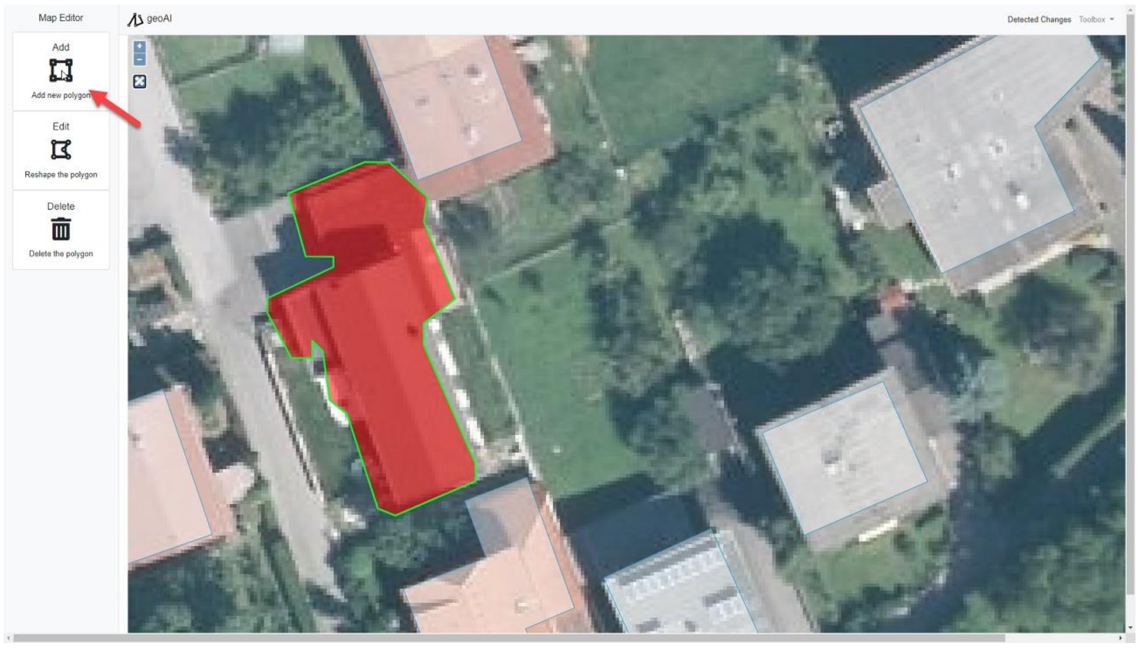

Figure 4.23. Adding detected building polygon to the user provided vector data (Can et al., 2021).

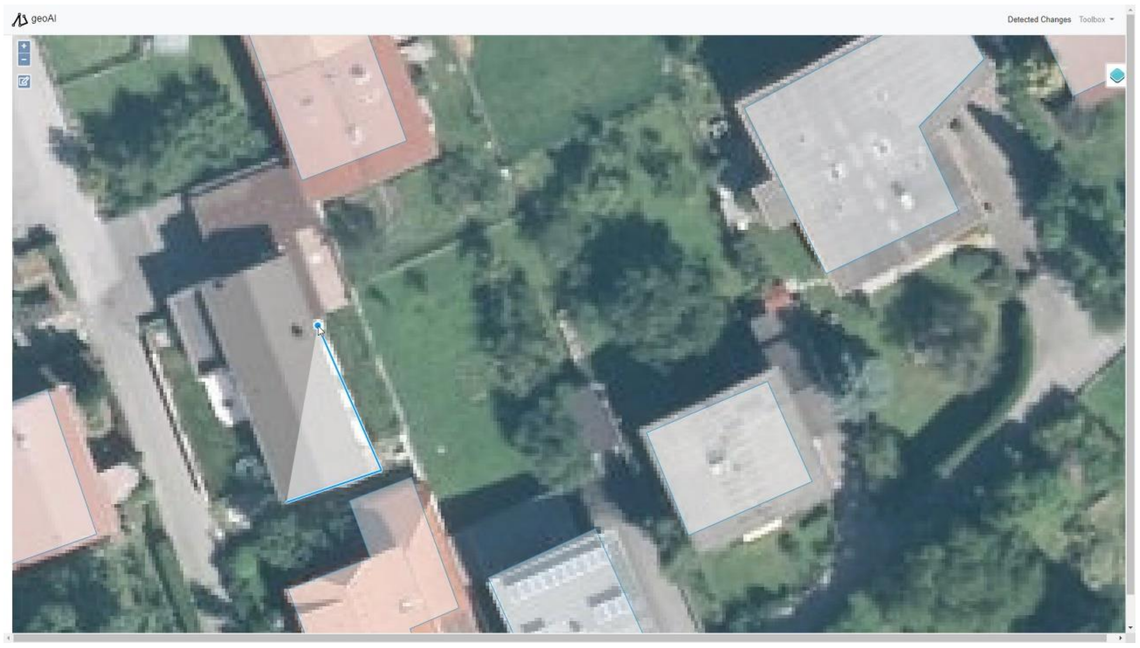

Figure 4.24. Drawing functionality of web map interface (Can et al., 2021).

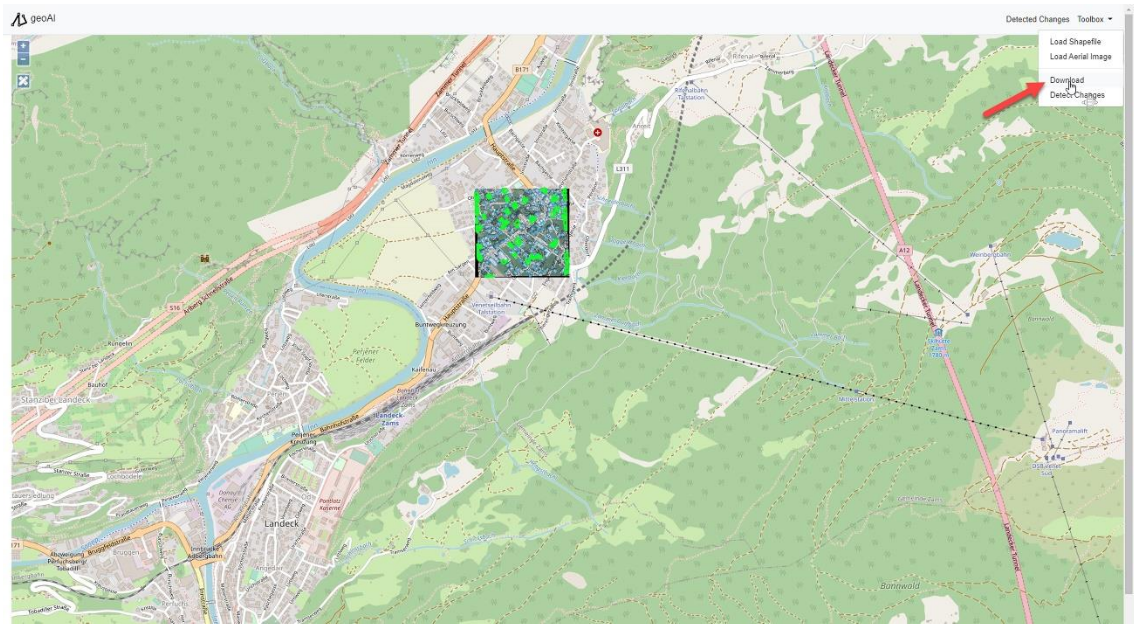

Figure 4.25. Download option for the updated vector layer (Can et al., 2021).

5. DISCUSSION

The aim of this thesis is to illustrate how the DL methods can help with different geospatial data processing and analysis tasks, such as change detection and map updating.

Due to the complexity of the buildings in terms of shape, size, diversity in roof materials, automatic extraction of building footprints is a major challenge. There exists several studies for extracting building footprints based on both the DL and the conventional approaches (Ok et al., 2013; Belgiu and Dragut, 2014; Huang et al., 2014; Bischke et al., 2017; Huang et al., 2017; Chen et al., 2018; Xu et al., 2018; Sun et al., 2018; Shrestha et al., 2018; Yuan, 2018; Liu et al., 2019; Shao et al., 2020; Guo et al., 2021; Xu et al., 2022). Developing a system that utilizes building footprint extraction operation requires well defined configurations and limitations. However, the framework proposed here was designed and developed as simply as possible in order to concentrate on the overall structure. The proposed framework may be customized for specific tasks and tailored to certain environments with various requirements. The DL model is still at the center of the framework, and several DL models can be used based on the challenge and the datasets available. In order to train and fine tune the DL models for improving the prediction performance, the models require well prepared datasets and significant computing resources, especially for aerial image segmentation tasks. Alternatively, synthetic building datasets produced by GAN-based approaches proposed by Chen et al. (2022) can be used for model training when well-prepared datasets are not available. If computing resources are restricted, the model training from scratch could take many days to months with no guarantee of the final model's or configuration's success. The model training is an iterative process in which hyper-parameters need to be fine-tuned, and the model needs to be modified until the desired performance is obtained.

In this thesis, a DL architecture adapted from medical image processing domain was utilized, and the model was modified in order to employ airborne images. Since the computing resources were limited, a subset of the Inria Image labelling dataset (ca. 20% with 36 images for training and 10 images for testing) was used in this thesis to train a base model. Since the Inria Image Labelling dataset includes aerial orthorectified images, the proposed framework was only tested with the images of such geometry. The model is expected to perform better when true-ortho images are provided. For off nadir images without the true-ortho processing, it is necessary to fine tune the model or utilize a new DL architecture, such as those proposed by Wang et al. (2022) and Esfandiari et al. (2021).

The model performance obtained here was not optimal since only a base DL model was trained for a few epochs with a fraction of the Inria Image Labelling dataset, and the changes observed suffered from erroneous segmentation output. Since the main focus was the development of the overall structure of the WebGIS, the comprehensive assessment of building footprint extraction by the DL model was left for future work. However, when deploying such a system, an evaluation method should be included for assessing the system’s performance. Evaluation methods including matched rates, shape similarity, and positional accuracy metrics for extracted buildings were reviewed by Zeng et al. (2013), and the authors proposed a new evaluation system for the accuracy assessment of building extraction. Various approaches have been proposed to assess building footprint data (Arkin et al., 1991; Zhan et al., 2005; Avbelj et al., 2015; Xu et al., 2017). As a result, improving the DL model to increase the prediction performance and assessing the building footprint predictions are still works in progress. In addition, as previously indicated, alternative DL models developed especially for building footprint extraction and change detection tasks (i.e., Zheng et al., 2022; Qian et al., 2022; Wei et al., 2022; Xu et al., 2022; Gao et al., 2022; Guo et al., 2021; Das and Chand, 2021; Li and Xin, 2021; Zorzi el al., 2020; Yang et al., 2020; Zorzi and Fraundorfer, 2019; Liu et al., 2019) may be added into the framework.

Even if the DL model included in the framework is a well-trained model with high accuracy, there may be cases where it does not perform well due to variations in the user provided raster data. In such cases, the development of solutions that allow users to fine-tune the DL model may contribute to the reduction of manual intervention. The updates made by the user in the regions where a change is detected can be further utilized as ground truth and a training dataset can be created from this data as well. After the dataset is created, the DL model can be fine-tuned with this dataset. This process can provide better performance of the DL model for the corresponding region, but if the system is to be used on a global scale, it should be tested whether this process impairs the generalizability of the DL model.

Utilizing the proposed framework on a global scale can be costly. In order to determine whether there is a change, first the raster data is passed through the DL model, then the model outputs are vectorized and then compared with the vector data provided by the user. It can be questioned how effective it is to perform these operations with very high resolution aerial images on a global scale. Instead, multi-scale solutions that can extract potential change areas from lower resolution satellite images, i.e. Sentinel-2, can be added to the proposed system as an intermediate processing component. As an example, the last modification date of the vector data and the creation date of the raster data can be requested from the user and the bounding boxes of the potential areas where the changes occur can be determined from the multi-temporal satellite images covering these dates. Instead of giving all very high resolution images as input to the framework, only the tiles containing the corresponding bounding box can be input to the system. Both the cost that will arise when the proposed system is operated on a global scale and how much this cost can be improved with intermediate hierarchical solutions are left to future studies.

As stated by Guo et al. (2020), buildings of varying sizes, color, and roof materials make it difficult to separate significant aspects from extraneous ones in VHR imagery. Therefore, when feeding into the change detection component, the input raster image resolution must also be considered. Other factors influencing the success of change detection include image acquisition angles such as nadir and off-nadir, atmospheric conditions, image acquisition date, and the geographic location of the images. Errors in both georeferencing information and co-registration of the input data would also distort the results and need to be taken into account as an additional processing task.

Given the completeness and scalability of the developed framework, it might be good to practice for additional optimization enhancements of the framework and handle the security related issues before deployment by mapping agencies or geospatial enterprises. However, if the framework is to be deployed on a worldwide basis, data input-output standards must also be addressed since they may pose significant challenges. In addition, the performance considerations such as the number of concurrent users running the change detection task or modifying the vector data, as well as the number of requests that the servers can manage, must be taken into account by the relevant agencies/enterprises.

6. CONCLUSION AND FUTURE WORK

To demonstrate how DL can facilitate geodatabase updating for building rooftop boundaries, in this thesis, a DL-assisted WebGIS framework was designed and implemented. The proposed framework has the potential to be implemented in a wide range of geospatial applications, especially those in which rapid data processing is needed or where data quality, particularly completeness, is essential. The open WebGIS platforms such as OSM or the ones from mapping agencies as well as geospatial firms may utilize the proposed framework in their image processing pipelines to improve the geodatabase updating tasks.

The open source tools and software utilized here involve JavaScript, HTML, CSS, Openlayers, jQuery, Bootstrap, Django, scikit-image, Tensorflow, GDAL/OGR, GeoPandas, GeoAlchemy-2, Geoserver, PostgreSQL/PostGIS and Apache Tomcat. The selection of the technologies was based on being open source, the familiarity of the developer, and the integration potential. In addition, the proposed framework will be integrated to GeoNode (GeoNode, 2022), an open source geospatial content management system including most of the technologies that were used in this thesis, i.e. Django, Geoserver and OpenLayers. The results have shown that the implementation could be carried out seamlessly and that the user interface and the functionalities developed for this purpose fulfill the user requirements. A major advantage of the developed system is the ability to adapt different DL models within the system. Thus, in future work, different DL models and further functionality can be added for various geospatial analysis purposes, such as land use and land cover classification, detection of particular geographic features such as water bodies, roads, tree species, etc. In addition, the DL models for building detection can be specialized for the different types of input data, such as multispectral band configurations, inclusion of DSM, UAV and satellite imagery, etc.

Thus, additional features, such as periodic monitoring and further change detection on the images after the determination of the areas that need to be updated, can be included as add-ons as future work. The proposed framework is considered to be connected to the GeoCitSci.com, a CitSci platform for geoscience investigations established by the

Geomatics and Geological Engineering Departments in collaboration at Hacettepe University (Kocaman and Gokceoglu, 2019; Can et al., 2019; Can et al., 2020). Consequently, the proposed framework may use additional DL algorithms for characterization of various geomorphological properties and geohazards.

REFERENCES

Abadi, M., Agarwal, A., Barham, P. et al., 2015. TensorFlow: Large-scale machine learning on heterogeneous systems. www.tensorflow.org (Accessed on: 25 April 2021)

Adam A., 2013. GIS Handbook for municipalities, United Nations Human Settlements Programme, UN-Habitat. https://unhabitat.org/sites/default/files/download-manager-files/GIS%20Handbook%20for%20Municipalities.pdf (Accessed on: 08 May 2022)

Agrawal, S., Gupta, R.D., 2017. Web GIS and its architecture: a review. *Arabian Journal of Geosciences*, 10(23), 1-13.

Antoniou, V. et al., 2017. The Future of VGI. In: Foody, G et al (eds.), Mapping and the Citizen Sensor. London: Ubiquity Press.

Apache, 2022. https://www.apache.org/ (Accessed on: 12 May 2022)

Arkin, E. M., Chew, L. P.,Huttenlocher, D. P.,Kedem, K.,Mitchell, J. S. B., 1991. An efficiently computable metric for comparing polygonal shapes. IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 13, no. 3, pp. 209-216, doi: 10.1109/34.75509.

Atasoy, K., Kocaman, S., 2021. Development Of A Geo-Analytics Platform For Post-Disaster Ground Assessment, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLIII-B3-2021, 685–692,.

Avbelj, J., Müller, R., Bamler, R., 2015. A Metric for Polygon Comparison and Building Extraction Evaluation. IEEE Geoscience and Remote Sensing Letters, vol. 12, no. 1, pp. 170-174, doi: 10.1109/LGRS.2014.2330695.

Balla, D., Zichar, M., Kiss, E., Szabó, G., Mester, T., 2022. Possibilities for Assessment and Geovisualization of Spatial and Temporal Water Quality Data Using a WebGIS Application. ISPRS Int. J. Geo-Inf., 11, 108.

Bayer, M., 2012. SQLAlchemy. In Amy Brown and Greg Wilson, editors, The Architecture of Open Source Applications Volume II: Structure, Scale, and a Few More Fearless Hacks.http://aosabook.org/en/index.html(Accessed on: 16 May 2022)

Belgiu, M., Drăguţ, L., 2014.Comparing supervised and unsupervised multiresolution segmentation approaches for extracting buildings from very high resolution imagery. ISPRS J. Photogramm. Remote Sens., 96, 67–75, doi: 10.1016/j.isprsjprs.2014.07.002.

Bing Maps, 2022. https://www.bingmapsportal.com/ (Accessed on: 14 May 2022)

Bischke, B.; Helber, P.; Folz, J.; Borth, D.; Dengel, A., 2017. Multi-Task Learning for Segmentation of Building Footprints with Deep Neural Networks. arXiv; arXiv:1709.05932.

Bittner, K., Adam, F., Cui, S., Körner, M., Reinartz, P., 2018. Building footprint extraction from VHR remote sensing images combined with normalized DSMs using fused fully convolutional networks. IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, 11(8), 2615-2629.

Bootstrap, 2021. https://getbootstrap.com/ (Accessed on: 25 April 2021).

Bradski, G., 2000. The OpenCV Library. Dr. Dobb's Journal of Software Tools.

Bulma, 2022. https://bulma.io/. (Accessed on: 12 May 2022)

Buyukdemircioglu M., Can R., Kocaman S., Kada M., 2022. Deep Learning Based Building Footprint Extraction From Very High Resolution True Orthophotos and nDSM. ISPRS Congress 2022 (Accepted for publication)

Can, R., Kocaman, S., Gokceoglu, C., 2019. A Convolutional Neural Network Architecture for Auto-Detection of Landslide Photographs to Assess Citizen Science and Volunteered Geographic Information Data Quality. *ISPRS International Journal of Geo-Information*, 8(7), 300.

Can, R., Kocaman, S., Gokceoglu, C., 2020. Development Of A CitSci And Artificial Intelligence Supported GIS Platform For Landslide Data Collection. *Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci*., 43, 43-50.

Cesium, 2022. https://cesium.com/platform/cesiumjs/ (Accessed on: 12 May 2022)

Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A. L., 2018. DeepLab: Semantic Image Segmentation With Deep Convolutional Nets, Atrous Convolution, And Fully Connected CRFS. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 40 (4), 834–848.

Chen, J., Zipf, A., 2017. DeepVGI: Deep Learning With Volunteered Geographic Information. In Proceedings of the 26th International Conference on World Wide Web Companion, April, pp. 771-772.

Chen, H., Li, W., Shi, Z., 2022. Adversarial Instance Augmentation for Building Change Detection in Remote Sensing Images. IEEE Transactions on Geoscience and Remote Sensing, vol. 60, pp. 1-16,doi: 10.1109/TGRS.2021.3066802.

Chen, R.X., Li, X.H., Li, J., 2018. Object-Based Features for House Detection from RGB High-Resolution Images. Remote Sens.-Basel, 10, 451.

Chen, T. et al., 2015. MXNet: A Flexible And Efficient Machine Learning Library For Heterogeneous Distributed Systems. http://www.arxiv.org/abs/1512.01274 (Accessed on : 16 May 2022)

Das, P. and Chand, S. AttentionBuildNet for Building Extraction from Aerial Imagery. 2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS), pp. 576-580, doi: 10.1109/ICCCIS51004.2021.9397178.

de Sherbinin, A. et al., 2021. The critical importance of citizen science data. Frontiers in Climate. 3, 2624-9553.
Django, 2022. https://www.djangoproject.com/ (Accessed on:12 May 2022)

ECSA (European Citizen Science Association), 2015. Ten Principles of Citizen Science. Berlin.

Elmasri, R., Navathe, S.,2015. Fundamentals of data base systems (7th ed.). Pearson: Addison Wesley

ESA SNAP, 2022. https://senbox.atlassian.net/wiki/spaces/SNAP/pages/19300362/How+to+use+the+SNAP+API+from+Python (Accessed on: 12 May 2022)

Esfandiari, M., Abdi, G., Jabari, S., Lolla, V. S. P., 2021. Building Change Detection in Off-Nadir Images Using Deep Learning. 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, pp. 1347-1350, doi: 10.1109/IGARSS47720.2021.9553172.

ESRI, 2022. https://developers.arcgis.com/python/ (Accessed on:12 May 2022)

Fan, H., Kong, G., Zhang, C. ,2021. An Interactive platform for low-cost 3D building modeling from VGI data using convolutional neural network. Big Earth Data, 5(1), 49-65.

FastAPI, 2022. FastAPI Framework. https://fastapi.tiangolo.com/ (Accessed on: 14 May 2022)

Flask, 2022. https://flask.palletsprojects.com/en/2.1.x/ (Accessed on: 12 May 2022)

Foundation, 2022. https://get.foundation/index.html (Accessed on: 12 May 2022)

Gao, S., Li, W., Sun, K., Wei, J., Chen, Y., Wang, X., 2022. Built-Up Area Change Detection Using Multi-Task Network with Object-Level Refinement. Remote Sensing, 14:4,957,doi:10.3390/rs14040957.

GDAL/OGR contributors, 2021. GDAL/OGR Geospatial Data Abstraction Software Library. Open Source Geospatial Foundation. https://gdal.org (Accessed on: 25 April 2021)

GeoAlchemy 2, 2022. https://geoalchemy-2.readthedocs.io/en/latest/ (Accessed on: 12 May 2022)

GeoDjango, 2022. https://docs.djangoproject.com/en/4.0/ref/contrib/gis/ (Accessed on:12 May 2022)

GeoNode, 2022. https://geonode.org/ (Accessed on: 19 July 2022)

Geoserver, 2021. http://geoserver.org/ (Accessed on: 25 April 2021)

Goodchild, M.F.,2001. Geographic Information Systems, International Encyclopedia of the Social & Behavioral Sciences, 6175-6182.

Goodchild, M.F., 2007. Citizens as sensors: the world of volunteered geography. GeoJournal 69, 211–221.

Goodfellow et al., 2016. Deep Learning, MIT Press. http://www.deeplearningbook.org (Accessed on: 14 May 2022)

Google Earth Engine, 2022. https://developers.google.com/earth-engine/guides/python_install (Accessed on: 12 May 2022)

Google Maps, 2022. https://developers.google.com/maps (Accessed on: 12 May 2022)

Guo, H., Shi, Q., Marinoni, A., Du, B., Zhang, L., 2021.Deep building footprint update network: A semi-supervised method for updating existing building footprint from bi-temporal remote sensing images. Remote Sens. Environ., 264, p. 112589,doi: 10.1016/j.rse.2021.112589

Hazirbas, C., Ma, L., Domokos, C., Cremers, D., 2017. FuseNet: Incorporating Depth into Semantic Segmentation via Fusion-Based CNN Architecture. Computer Vision – ACCV 2016, volume 10111, pages 213–228. Springer International Publishing, Cham. Series Title: Lecture Notes in Computer Science.

He, K., Gkioxari, G., Dollár, P., Girshick, R., 2017. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision, 2961-2969.

Hexagon, 2022. https://www.hexagongeospatial.com/products/power-portfolio/erdas-imagine (Accessed on: 14 May 2022)

Huang, X.; Yuan, W.; Li, J.; Zhang, L., 2017. A New Building Extraction Postprocessing Framework for High-Spatial-Resolution Remote-Sensing Imagery. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens. vol. 10, no. 2, pp. 654-668,doi: 10.1109/JSTARS.2016.2587324.

Huang, X.; Zhang, L.; Zhu, T., 2014. Building change detection from multitemporal high-resolution remotely sensed images based on a morphological building index. IEEE J. Sel. Top. Appl. Earth Obs. Remote Sens., 7, 105–115, doi: 10.1109/JSTARS.2013.2252423.

Huisman, O., de By, R. A., 2009. Principles of geographic information systems : an introductory textbook. ITC Educational Textbook Series; International Institute for Geo-Information Science and Earth Observation, 1-17. http://www.itc.nl/library/papers_2009/general/PrinciplesGIS.pdf (Accessed on: 08 May 2022)

Janowicz, K., Gao, S., McKenzie, G., Hu, Y. and Bhaduri, B., 2020. GeoAI: spatially explicit artificial intelligence techniques for geographic knowledge discovery and beyond. International Journal of Geographical Information Science, 34(4), pp.625-636.

JavaScript, 2022. https://www.javascript.com/ (Accessed on:12 May 2022)

Jia Y., Shelhamer E., Donahue J., Karayev S., Long J., Girshick R., Guadarrama S., and Darrell T., 2014. Caffe: Convolutional Architecture for Fast Feature Embedding. In Proceedings of the 22nd ACM international conference on Multimedia (MM '14). Association for Computing Machinery, New York, NY, USA, 675–678.

Jiwani, A., Ganguly, S., Ding, C., Zhou, N., Chan, D. M., 2021. A Semantic Segmentation Network for Urban-Scale Building Footprint Extraction Using RGB Satellite Imagery. arXiv preprint. https://arxiv.org/abs/2104.01263 (Accessed on: 16 May 2022)

jQuery, 2022. https://jquery.com/ (Accessed on: 12 May 2022)

Kada, M. and Kuramin, D., 2021. ALS Point Cloud Classification Using Pointnet++ And Kpconv with Prior Knowledge, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLVI-4/W4-2021, 91–96.

Kelsey Jordahl, Joris Van den Bossche, Martin Fleischmann, Jacob Wasserman, James McBride, Jeffrey Gerard, … François Leblanc. (2020, July 15). geopandas/geopandas: v0.8.1 (Version v0.8.1). Zenodo.

Kocaman, S., Gokceoglu, C., 2018. On the use of Citsci and VGI in Natural Hazard Assessment. The International Archives of Photogrammetry, Remote Sensing and Spatial Information Sciences, 42, 69-73.

Kocaman, S. Gokceoglu, C., 2019. A CitSci app for landslide data collection. *Landslides*, 16, 611.

L3Harris Geospatial, 2022. https://envi-py-engine.readthedocs.io/en/latest/ (Accessed on: 12 May 2022)

La Guardia, M.; D'Ippolito, F.; Cellura, M., 2021. Construction of a WebGIS Tool Based on a GIS Semiautomated Processing for the Localization of P2G Plants in Sicily (Italy). ISPRS Int. J. Geo-Inf., 10, 671.

Leaflet, 2022. https://leafletjs.com/ (Accessed on: 12 May 2022)

LeCun, Y., Bengio, Y., Hinton, G., 2015. Deep learning. Nature 521, 436–444.

Li, Z., Wegner, J. D., Lucchi, A., 2019. Topological map extraction from overhead images. In Proceedings of the IEEE/CVF International Conference on Computer Vision (pp. 1715-1724).

Li, H., Herfort, B., Lautenbach, S., Chen, J. Zipf, A. ,2022. Improving OpenStreetMap missing building detection using few-shot transfer learning in sub-Saharan Africa. Transactions in GIS, 00, 1– 22.

Li, Z. and Xin, Q., 2021.Corner-Guided Building Polygon Construction from Aerial Images using Deep Multitask Learning.2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS, pp. 4043-4046, doi: 10.1109/IGARSS47720.2021.9554624.

Li, W. ,2020. GeoAI: Where machine learning and big data converge in GIScience. Journal of Spatial Information Science, 20, 71-77.

Li, Z.; Xin, Q.; Sun, Y.; Cao, M., 2021. A Deep Learning-Based Framework for Automated Extraction of Building Footprint Polygons from Very High-Resolution Aerial Imagery. Remote Sensing, 13, 3630.

Liu, P., Liu, X., Liu, M., Shi, Q., Yang, J., Xu, X., Zhang, Y.,2019. Building Footprint Extraction from High-Resolution Images via Spatial Residual Inception Convolutional Neural Network. Remote Sens., 11, 830. doi:10.3390/rs11070830

Maggiori E., Tarabalka Y., Charpiat G., Alliez P., 2017. “Can Semantic Labeling Methods Generalize to Any City? The Inria Aerial Image Labeling Benchmark”. IEEE International Geoscience and Remote Sensing Symposium (IGARSS)

Mapbox, 2022. https://www.mapbox.com/ (Accessed on: 12 May 2022)

MapServer, 2022. https://mapserver.org/ (Accessed on: 12 May 2022)

MariaDB, 2022. https://mariadb.org/ (Accessed on: 12 May 2022)

Marmanis, D., Schindler, K., Wegner, J. D., Galliani, S., Datcu, M., & Stilla, U., 2018. Classification with an edge: Improving semantic image segmentation with boundary detection. ISPRS Journal of Photogrammetry and Remote Sensing, 135, 158-172.

Masonite, 2022. Masonite Project. https://docs.masoniteproject.com/ (Accessed on: 14 May 2022)

Microsoft, 2021. Bing Maps APIs. https://www.microsoft.com/en-us/maps/choose-your-bing-maps-api (Accessed on 24.04.2021)

MongoDB, 2022. https://www.mongodb.com/ (Accessed on:12 May 2022)

Moreri, K., Fairbairn, D., James, P. , 2018. Issues in developing a fit for purpose system for incorporating VGI in land administration in Botswana. Land Use Policy, 77, 402-411.

MySQL, 2022. https://www.mysql.com/ (Accessed on: 12 May 2022)

NGINX, 2022. https://www.nginx.com/ (Accessed on: 12 May 2022)

Nielsen, M. A., 2015. Neural Networks and Deep Learning. Determination Press

Nunes, L.J.R.; Curado, A.; Azevedo, R.; Silva, J.P.; Lopes, N.; Lopes, S.I., 2022. Designing a Multicriteria WebGIS-Based Pre-Diagnosis Tool for Indoor Radon Potential Assessment. Appl. Sci.,12, 1412.

OGC GML, 2022. Geography Markup Language. https://www.ogc.org/standards/gml (Accessed on: 14 May 2022)

OGC Standards, 2022. https://www.ogc.org/docs/is (Accessed on: 14 May 2022)

OGC WFS, Web Feature Service. https://www.ogc.org/standards/wfs (Accessed on:14 May 2022)

Ok, A.O, Senaras C., Yuksel B., 2013. Automated Detection of Arbitrarily Shaped Buildings in Complex Environments From Monocular VHR Optical Satellite Imagery. IEEE Transactions on Geoscience and Remote Sensing, vol. 51, no. 3, pp. 1701-1717, March 2013, doi: 10.1109/TGRS.2012.2207123.

Oluwatosin, H. S., 2014. Client-server model. IOSR Journal of Computer Engineering, 16(1):67–71, ISSN 2278-8727

ol-layerswitcher, 2022. https://github.com/walkermatt/ol-layerswitcher (Accessed on: 12 May 2022)

OpenLayers, 2021. https://openlayers.org/ (Accessed on: 25 April 2021).

Open Street Map, 2022. OpenStreetMap Project Contributors. https://wiki.openstreetmap.org/wiki/Contributors (Accessed on : 31 May 2022).

Ostermann, F. O., Granell, C., 2017. Advancing science with VGI: Reproducibility and replicability of recent studies using VGI. Transactions in GIS, 21(2), 224-237.

Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., … Chintala, S., 2019. PyTorch: An Imperative Style, High-Performance Deep Learning Library. In Advances in Neural Information Processing Systems 32 (pp. 8024–8035). Curran Associates, Inc. http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf (Accessed on: 16 May 2022)

Patera, A.; Pataki, Z.; Kitsiou, D. Development of a webGIS Application to Assess Conflicting Activities in the Framework of Marine Spatial Planning, 2022.J. Mar. Sci. Eng. 10, 389.

PCI Geomatics, 2022. https://catalyst.earth/catalyst-system-files/api-doc/sphinx/html/ (Accessed on: 12 May 2022)

PostgreSQL, 2022. https://www.postgresql.org/ (Accessed on: 12 May 2022)

Postgis, 2022. https://postgis.net/ (Accessed on: 12 May 2022)

Precisely, 2022. https://www.precisely.com/product/precisely-mapinfo/mapinfo-pro (Accessed on: 14 May 2022)

Pyscopg, 2022. https://www.psycopg.org/ (Accessed on: 12 May 2022)

Qian, Z., Chen, M., Zhong, T., Zhang, F., Zhu, R., Zhang, Z., Lü, G., 2022. Deep Roof Refiner: A detail-oriented deep learning network for refined delineation of roof structure lines using satellite imagery. International Journal of Applied Earth Observation and Geoinformation, 107, 102680, doi:10.1016/j.jag.2022.102680.

QGIS, 2022. https://qgis.org/pyqgis/3.0/ (Accessed on: 12 May 2022)

Ronneberger, O., Fischer, P., Brox, T., 2015. U-net: convolutional networks for biomedical image segmentation. In Medical image computing and computer-assisted intervention (MICCAI), volume 9351 of LNCS. New York, NY:Springer,, 234–241. Available on arXiv:1505.04597 [cs.CV].

Rowland, A., Folmer, E., Beek, W., 2020. Towards Self-Service GIS—Combining the Best of the Semantic Web and Web GIS. ISPRS International Journal of Geo-Information, 9(12), 753.

Safe, 2022. https://www.safe.com/ (Accessed on: 14 May 2022)

Sahu, M. Ohri, A., 2019. Vector Map Generation from Aerial Imagery Using Deep Learning. *ISPRS Ann. Photogramm. Remote Sens. Spatial Inf. Sci.*, IV-2/W5, 157-162.

Sanic, 2022. Sanic Framework. https://sanic.readthedocs.io/en/stable/ (Accessed on: 14 May 2022)

See, L.; Mooney, P.; Foody, G.; Bastin, L.; Comber, A.; Estima, J.; Fritz, S.; Kerle, N.; Jiang, B.; Laakso, M.; Liu, H.-Y.; Milčinski, G.; Nikšič, M.; Painho, M.; Pődör, A.; Olteanu-Raimond, A.-M.; Rutzinger, M. Crowdsourcing, Citizen Science or Volunteered Geographic Information? The Current State of Crowdsourced Geographic Information. ISPRS International Journal of Geo-Information,2016, 5, 55.

Senzaki, M., Barber, J.R., Phillips, J.N. et al., 2020. Sensory pollutants alter bird phenology and fitness across a continent. Nature 587, 605–609.

Shao, Z., Tang, P., Wang, Z., Saleem, N., Yam, S., Sommai, C.,2020. BRRNet: A Fully Convolutional Neural Network for Automatic Building Extraction From High-Resolution Remote Sensing Images. Remote Sens., 12, 1050.doi:10.3390/rs12061050

Shrestha, S., Vanneschi, L.,2018. Improved fully convolutional network with conditional random fields for building extraction. Remote Sens. Lett., 10, 1135.

Shi, Y., Li, Q., Zhu, X. X., 2018. Building footprint generation using improved generative adversarial networks. *IEEE Geoscience and Remote Sensing Letters*, 16(4), 603-607.

Software Using GDAL, 2022. https://gdal.org/software_using_gdal.html (Accessed on: 14 May 2022)

Sun, Y., Zhang, X., Zhao, X., Xin, Q., 2018. Extracting Building Boundaries from High Resolution Optical Images and LiDAR Data by Integrating the Convolutional Neural Network and the Active Contour Model. Remote Sens., 10, 1459. https://doi.org/10.3390/rs10091459

Sun, X., Zhao, W., Maretto, R. V., and Persello, C., 2021. Building Outline Extraction from Aerial Imagery and Digital Surface Model with A Frame Field Learning

Framework, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLIII-B2-2021, 487–493.

Taylor et al., 2021. A citizen science approach to identifying trace metal contamination risks in urban gardens. Environment International, 155.

Tensorflow Keras Applications, 2022. https://www.tensorflow.org/api_docs/python/tf/keras/applications (Accessed on: 14 May 2022)

The pandas development team, 2020. pandas-dev/pandas: Pandas.

Tornado, 2022. Tornado web framework. https://www.tornadoweb.org/en/stable/ (Accessed on: 14 May 2022)

Torchvision Models, 2022. https://pytorch.org/vision/stable/models.html (Accessed on: 14 May 2022)

Toro Herrera, J. F., Carrion, D., Brovelli, M. A., 2021. A Collaborative Platform For Water Quality Monitoring: Simile Webgis, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLIII-B4-2021, 201–207.

United Nations, Department of Economic and Social Affairs, Population Division. *World Urbanization Prospects: The 2018 Revision (ST/ESA/SER.A/420)*; United Nations: New York, NY, USA, 2019.

Virtanen, P., Gommers, R., Oliphant, T.E., Haberland, M. et al., 2020. SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17(3), 261-272.

Visvalingam, M.,Whyatt, J. D., 1990. The Douglas-Peucker algorithm for line simplification: re-evaluation through visualization. In Computer Graphics Forum (Vol. 9, No. 3, pp. 213-225). Oxford, UK: Blackwell Publishing Ltd.

Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J.D., Yager, N., Gouillart, E., Yu, T., scikit-image contributors, 2014. scikit-image: Image processing in Python.

Wu, G., Shao, X., Guo, Z., Chen, Q., Yuan, W., Shi, X., Xu, Y., 2018. Automatic Building Segmentation of Aerial Imagery Using Multi-Constraint Fully Convolutional Networks. *Remote Sensing*, 10(3), 407.

Xu Y, Wu L, Xie Z, Chen Z., 2018. Building Extraction in Very High Resolution Remote Sensing Imagery Using Deep Learning and Guided Filters. Remote Sensing. 2018; 10(1):144.

Xu, H., Zhu, P., Luo, X., Xie, T., Zhang, L., 2022.Extracting Buildings from Remote Sensing Images Using a Multitask Encoder-Decoder Network with Boundary Refinement. Remote Sens., 14, 564.doi:10.3390/rs14030564.

Xu, Y., Wu, L., Xie, Z., Chen, Z., 2018. Building extraction in very high resolution remote sensing imagery using deep learning and guided filters. Remote Sens.-Basel, 10, 144.

Xu, Y., Chen, Z., Xie, Z., Wu, L., 2017. Quality assessment of building footprint data using a deep autoencoder network. International Journal of Geographical Information Science, 31:10, 1929-1951, doi: 10.1080/13658816.2017.1341632.

Xu S., Ehlers M., 2022.Automatic detection of urban vacant land: An open-source approach for sustainable cities.Computers, Environment and Urban Systems, 91.

Yalcin, I.; Kocaman, S.; Gokceoglu, C. A CitSci Approach for Rapid Earthquake Intensity Mapping: A Case Study from Istanbul (Turkey). ISPRS Int. J. Geo-Inf. 2020, 9, 266.

Yang, H., Wu, P., Yao, X., Wu, Y., Wang, B., Xu, Y., 2018. Building extraction in very high resolution imagery by dense-attention networks. *Remote Sensing*, 10(11), 1768.

Yang, G., Zhang, Q., Zhang, G., 2020. EANet: Edge-Aware Network for the Extraction of Buildings from Aerial Images. Remote Sensing., 12:13, 2161, doi:10.3390/rs12132161.

Yi, Y.; Zhang, Z.; Zhang, W.; Zhang, C.; Li, W.; Zhao, T., 2019. Semantic Segmentation of Urban Buildings from VHR Remote Sensing Imagery Using a Deep Convolutional Neural Network. Remote Sensing. 2019; 11(15):1774.

Yuan, J., 2018. Learning building extraction in aerial scenes with convolutional networks. IEEE Trans. Pattern Anal., 40, 2793–2798.

Zeng, C., Wang, J., Lehrbass, B., 2013.An Evaluation System for Building Footprint Extraction From Remotely Sensed Data.IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 6, no. 3, pp. 1640-1652,doi: 10.1109/JSTARS.2013.2256882.

Zheng, H.,Gong, M.,Liu, T.,Jiang, F.,Zhan, T.,Lu, D.,Zhang, M., 2022.HFA-Net: high frequency attention siamese network for building change detection in VHR remote sensing images.Pattern Recognit., 129, p. 108717, doi:10.1016/j.patcog.2022.108717

Zhan, Q.M., Molenaar, M., Tempfli, K.,Shi, W.Z., 2005.Quality assessment for geo-spatial objects derived from remotely sensed data.International Journal of Remote Sensing, Vol. 26, 14, pp. 2953–2974, doi:10.1080/01431160500057764.

Zhao, W., Persello, C., & Stein, A., 2021. Building outline delineation: From aerial images to polygons with an improved end-to-end learning framework. ISPRS Journal of Photogrammetry and Remote Sensing, 175, 119-131.

Zorzi, S. and Fraundorfer, F., 2019.Regularization of Building Boundaries in Satellite Images Using Adversarial and Regularized Losses. IGARSS 2019 - 2019 IEEE International Geoscience and Remote Sensing Symposium, pp. 5140-5143, doi: 10.1109/IGARSS.2019.8900337.

Zorzi, S.,Bittner, K.,Fraundorfer, F., 2020. Map-Repair: Deep Cadastre Maps Alignment and Temporal Inconsistencies Fix in Satellite Images. IGARSS 2020 - 2020 IEEE International Geoscience and Remote Sensing Symposium, pp. 1829-1832, doi: 10.1109/IGARSS39084.2020.9323370.

Wang, J. , Meng, L., Li, W. , Yang, W., Yu, L., Xia, G. -S., 2022.Learning to Extract Building Footprints from Off-Nadir Aerial Images.IEEE Transactions on Pattern Analysis and Machine Intelligence, doi: 10.1109/TPAMI.2022.3162583.

web2py, 2022. Web2py Web Framework. http://web2py.com/ (Accessed on: 14 May 2022)

Wei, S., Zhang, T., Ji, S., 2022.A Concentric Loop Convolutional Neural Network for Manual Delineation-Level Building Boundary Segmentation From Remote-Sensing Images. IEEE Transactions on Geoscience and Remote Sensing, vol. 60, pp. 1-11, doi: 10.1109/TGRS.2021.3126704.

APPENDICES

APPENDIX A – Proceedings derived from the thesis

Can, R., Kocaman, S., and Ok, A. O.,2021. A WebGIS Framework For Semi-Automated Geodatabase Updating Assisted By Deep Learning, Int. Arch. Photogramm. Remote Sens. Spatial Inf. Sci., XLIII-B5-2021, 13–19, doi:10.5194/isprs-archives-XLIII-B5-2021-13-2021.