FACULTY
OF INFORMATION
TECHNOLOGY
CTU IN PRAGUE

Assignment of bachelor's thesis

Title:	Segmentation of historical maps by deep learning
Student:	Matěj Polák
Supervisor:	Mgr. Petr Šimánek
Study program:	Informatics
Branch / specialization:	Knowledge Engineering
Department:	Department of Applied Mathematics
Validity:	until the end of summer semester 2023/2024

Instructions

The goal of the thesis is to understand different ways in which we can apply deep learning to segment historical maps or detect specific features in the maps. Analysis of historical maps is very important for historians and analysis of urban and natural development and change in time.

1) Do a literature survey on current approaches to image segmentation.
2) Analyze a dataset of historic maps [1].
3) Choose and describe two methods (e.g. YOLO, Transformer, UNet, [2]) or suggest a new method.
4) Implement the methods in PyTorch, train and test the model.
4) Compare and analyze the results in detail.

[1] Generic Semantic Segmentation of Historical Maps, Rémi Petitpierre, et al.
[2] TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation, Jieneng Chen, et al.

Electronically approved by Ing. Magda Friedjungová, Ph.D. on 31 October 2022 in Prague.

Bachelor’s thesis

SEGMENTATION OF HISTORICAL MAPS BY DEEP LEARNING

Matěj Polák

Faculty of Information Technology
Katedra aplikované matematiky
Supervisor: Mgr. Petr Šimánek
February 15, 2024

Contents

List of Figures

List of Tables

I express my sincere thanks to my supervisor, Mgr. Petr Šimánek, for his support and excellent guidance on my thesis. I would also like to say thank you to the Datalab team for allowing me to use their computational resources for training the models of this work. And lastly, my warm thanks goes to my family, friends and colleagues who have provided me with nothing but strength and support.

Abstract

The topic of the bachelor's thesis is the use of deep learning in the task of semantic segmentation of historical maps. The theoretical part presents current approaches to image segmentation in the context of historical document analysis. It then introduces definitions from the fields of image processing, machine learning and applied mathematics. In the practical part, a dataset of historical maps is analyzed and used for implementing two different deep learning models (UNet and TransUNet). It is followed by the experimental section, in which the effects of various modifications of the models on their performance are observed. Finally, the results of the practical and experimental parts are analyzed in the Discussion section. The outputs of the thesis can further be applied in the study of acquiring knowledge from historical documents.

Keywords deep learning, image segmentation, semantic segmentation, historical maps, neural networks, Transformers, Python, PyTorch, UNet, TransUNet

Abstrakt

Tématem bakalářské práce je použití metod hlubokého učení na úlohu sémantické segmentace historických map. Teoretická část uvádí současné přístupy k segmentace obrazu v kontextu analýzy historických dokumentů. Dále zavádí definice z oblasti zpracování obrazu, strojového učení a aplikované matematiky. V praktické části je analyzován konkrétní dataset historických map a použit pro implementaci dvou modelů hlubokého učení (UNet a TransUNet). Následuje experimentální sekce, v níž jsou sledovány vlivy dílčích modifikací modelů na jejich výkonnost. Výsledky praktické a experimentální části jsou analyzovány v Diskusi. Výstupy práce mohou být dále použity pro studium získávání znalostí z historických dokumentů.

Klíčová slova hluboké učení, segmentace obrazu, sémantická segmentace, historické mapy, neuronové sítě, transformery, Python, PyTorch, UNet, TransUNet

Introduction

The study of historical maps – and historical documents in general – plays a vital role in understanding our past. Historical maps are valuable resources for studying how urbanization, industrialization, historical events and other factors have altered the landscape. Comparing historical maps to contemporary ones provides an insight into how cities, countries and lands have changed over time.

Analyzing historical maps can be a difficult task. Some of the problems that need to be tackled in the analysis are: damaged/incomplete material, the lack of clarity in the author's intentions, unknown origin of the map, scarcity of other sources to compare with, missing/incomplete legend/scale and others. This is especially true for the older periods, when maps rarely had standards and the depiction of the landscape was up to the author.

For these reasons, people are unable to properly interpret all available materials and they are therefore turning their attention towards exploiting the modern technology. Most current approaches involve the use of neural networks – convolutional neural networks (CNNs) in particular. I find this encounter of historical artifacts and latest research in computer science very intriguing. Even more so with the hope that this connection between the old and the new will lead towards deeper understanding of the history.

Complete digital analysis of a historical document is a very complex task. This thesis will focus on one subtask in particular – the semantic segmentation (in other words: dividing the map up into several pre-defined classes or categories).

In the theoretical part, I will first introduce theory related to cartography, image processing and image segmentation. Next, I will present current approaches to segmentation of historical maps. I will conclude the part by describing the technology used in the practical part – CNNs, Transformers and the UNet and TransUNet architectures, in particular.

In the practical part, I will start by analyzing the dataset I will be focusing on. Next, I will build two models to try to solve the task of semantic segmentation. First, a UNet architecture, which is a convolutional neural network first developed for biomedical image segmentation. Then, I will try an alternative approach using TransUNet, a CNN-Transformer hybrid architecture. After implementing those, I will proceed to the experimental part, where I will study the effects of various modifications and alterations to the model. The part will finish by comparing the two approaches in the Discussion section.

Aims and Objectives

The goal of the thesis is to explore how well deep learning models can perform in the task of semantic segmentation of historical maps. This goal will be met by the means of the following objectives:

1. Present current approaches to image segmentation on historical maps and introduce relevant theory.
2. Implement two models (UNet, TransUNet) for semantic segmentation.
3. Carry out experiments on the models and describe how their efficiency is affected by various modifications.
4. Sum up the results of the experiment and compare the two models.

The outputs of the thesis can further be applied in the study of historical maps, and image segmentation in general. They could serve as a recommendation to researchers as to which approaches yield better performance.

Chapter 1

Theoretical part/Literature review

In this chapter, I will first introduce cartography and image segmentation techniques. Then, I will present current approaches to historical map segmentation in the literature review. After presenting this work's approach, I will conclude the chapter by listing the definitions necessary for the practical part.

1.1 Cartography

"Cartography is the art, science and technology of creating maps, as well as the study thereof as scientific documents and works of art."[1] In the broader sense, cartography includes studying the history of cartography, maintaining map collections, as well as the collection, manipulation and the design of maps.

Earliest known maps come from the prehistoric era, but it was Ancient Greeks and Romans who became the first true mapmakers thanks to their growing knowledge in mathematics, geometry and astronomy. The Romans were also motivated by the need of representing the network of roads (viae) that was essential for the prosperity of their imperium. During the Middle Ages, the so-called *mappae mundi* (maps of the world) were produced. Rather than navigation, they were used as teaching aids or for artistic purposes. The invention of the magnetic compass, telescope and sextant during the Age of Discovery made it possible to produce more accurate maps for nautical navigation. This was also the time when Flemish mapmaker Gerardus Mercator first published a map based on the Mercator projection, widely in use until today.

Further technological advances allowed for a more accurate representation and mass production on a larger scale. Maps became increasingly more needed for urban planning, transportation and trade, and for military purposes, it was especially crucial to have an accurate projection of the land.[2]

Cartography has two main functions: maps serve as historic sources, and they have a methodological function. Cartography is connected to many other fields of study: historiography, historic geography, archaeology, ethnography, art history, calligraphy, cartography history etc.[1]

Maps are a reduced, abstracted, generalized image of parts of the Earth, transcribed onto a plane using mathematically defined transformations – cartographic projections. There are many different ways of categorizing maps – based on their purpose, function, theme, subject matter, scale, symbolism, form, appearance, size, period of creation, country of origin etc. However, all maps are concerned with two elements of reality: locations and attributes. From these two basic elements, various relationships can be formed.

Similarly to literary sources, various features are studied for analysis of maps. They're authorship, period and place of map creation, fonts, material, art technique, scale, frame, legend and other. For analysis, knowledge from other disciplines are applied: auxiliary sciences of history, physics, chemistry, and recently also computation technology.[2]

1.2 Image segmentation

A digital image is a 2D image represented by a two-dimensional array of intensity samples – pixels. In the case of a color image, a vector of three elements makes up each pixel.

Image segmentation is the ”partition of an image into a set of regions that cover it”. A special type of image segmentation is semantic segmentation, which is the task of assigning each pixel with one or more (in the case of multi-class segmentation) categories (classes). This is different from instance segmentation, for which we differentiate between specific belonging instances of the object.[3]

Image segmentation techniques can be generally be divided into two major groups: classic segmentation methods and deep learning methods.[4] Here, I give the review of some of the most popular ones.

1.2.1 Classic methods

One of the most straightforward methods is thresholding. Lower and/or upper threshold intensity values are selected which binarize the image. The threshold values can be set manually, derived from the histogram or computed using such algorithms as Otsu's method[5].

Rule-based methods such as color thresholding are very straightforward, easy to implement and don't have any special computational requirements. However, Mäyrä et al.[6] claim that such methods are usually not suitable for the task of image segmentation of historic maps. Colors often overlap and mix into another color, the same color can have multiple meanings in a map and sometimes texture is just as important as color. Because of this, it is very challenging to find a single threshold and effectively segment the image.

Region growing is a method used in image processing to partition an image into coherent regions based on predefined criteria, starting from initial seed points and iteratively merging neighboring pixels or regions that meet certain similarity conditions. The watershed algorithm, on the other hand, treats an image as a topographic surface and simulates the flooding of basins from local minima, dividing the image into regions based on the dynamics of water flow.

Edge and corner detection is another useful technique. Areas where intensity levels change abruptly can be considered the boundaries of different regions. The derivative of the intensity level, obtained by the difference approximation, is used to identify these boundary changes. Edge detection yields different results based on the differential operator used. The most common operator choices are Canny, Sobel, Roberts and Laplacian. However, these operators are sensitive to noise and are only suitable for images with low noise and complexity.

K-means clustering is an iterative method used to partition an image into a pre-defined number of clusters by minimizing the sum of distances between data points and their respective cluster centroids. It iteratively assigns data points to the nearest centroid and updates centroids until convergence, producing clusters with similar data points grouped together. An alternative is the Mean Shift algorithm, which does not require the user to know the number of clusters beforehand.[4]

1.2.2 Deep learning methods

Deep learning is a field of machine learning where a model is trained using a layered hierarchy of concepts, enabling the computer to ”learn complicated concepts by building them out of simpler ones” [7]. This is achieved via use of artificial neural networks: convolutional neural networks, recurrent neural networks, transformers and others. Nowadays, deep learning has a wide range of applications, from computer vision and natural language processing to bioinformatics and medical image analysis.

A very popular types of deep learning models are of the encoder-decoder architecture. It consists of two parts: the encoder (contracting) part extracts relevant features from images and the decoder (expansive) part uses the extracted features to reconstruct a segmentation mask. The process of mapping back the multi-level features extracted by the encoder to the original image is called upsampling.[4]

Skip connections were developed to improve rough pixel positioning. It is a defining feature of UNet[8], where the skip connections are used to obtain details of images even in the decoder phase.

The attention mechanism allows to represent the dependency between different regions of an image, even long-distance relationships. The methods first developed for natural language processing are being successfully applied to computer vision as well[9]. The attention mechanism is also combined with existing approaches, bringing about hybrid architectures like AttUNet[10] and TransUNet[11].

Training image segmentation models has high computational costs, and creating suitable sample–label pairs by manual annotation is labor-intensive. In the context of segmentation of historical maps, however, Mäyrä et al. [6] claim that the use of deep learning models has advantages over classic methods: given enough training data, the model can learn to ignore textures that are not of interest, such as written text, property and municipal boundaries, and height contours.

The two classes of approaches – classic computer vision and deep learning – can be combined. Typically, while using a deep learning model such as the CNN, the images are preprocessed and postprocessed using various classic computer vision techniques, such as thresholding, morphological operations and other[12, 13].

1.3 State of the art

In the recent years, the amount of digitized historical maps that are available has grown rapidly [14]. This has led to an increased researchers’ interest in obtaining useful information from the data, using various methods.

R. Pétitpierre is the author of the dataset on which this work is focused. In a segmentation task, Pétitpierre et al. [15] present a method that measures the diversity of cartographic figuration in a map corpus. The model is a CNN with UNet architecture and ResNet pretrained on ImageNet as encoder and cross validation is used for evaluation. On the two corpora the dataset consists of, they achieved IoU of 0.89 and 0.80 for the 2+1 ontology, and 0.63 and 0.55 for the 4+1 ontology. Transfer learning, when one corpus is pretrained on the other, also showed minor improvement. The authors note that the model generally performs better on occidental maps. They also note that the removal of color has very low impact on the performance of the model.

Chazalon et al. [13] presented three different tasks as part of the ICDAR 2021 Competition on Historical Map Segmentation. The dataset in question is similar to the one introduced by Pétitpierre. In the first task, participants had to detect building blocks in the maps. The winning team used a weakly-supervised DenseNet-121-like model and various postprocessing techniques for binary segmentation of the maps. The second task consists of segmenting the map content from the rest of the sheet. The winning method of this task was a UNet-based model, followed by a recursive Otsu filter. In the third task, participants were challenged to locate the intersections of graticule lines. The winning method is rule-based and does not require any training or annotated data: it first generates a binary image using a recursive Otsu filter, then, graticules are detected using a Hough Line Transform, and finally the intersections are estimated from the Hough line intersections. In conclusion, while task 1 needs some progress to be fully automated, tasks 2 and 3 are ”almost solved” by the proposed approaches.

Mäyrä et al. [6] used a modified UNet model to analyze changes in the land cover of Southern Finland between the years 1965, 1985, 2005 and 2022. To extract information from the maps, they chose five target classes: fields, mires, roads, watercourses and water bodies. Multiple preprocessing and postprocessing steps were applied on the maps covering the study area: cropping, color adjusting, georeferencing, morphological opening, linearization and other. UNet with a pretrained ResNet152 encoder was employed to train on the data and the model yielded ”excellent” results, with test IoU of 0.783 and F1 of 0.872.

Ekim et al. [16] used a similar UNet-based method for detecting and classifying different types of roads from German maps of Turkey from World War II. Six segmentation classes were laid down: stabilized roads, cart roads, inferior roads, railways, footpaths and the background class. They used the UNet++ architecture with ResneXt50_32x4d as its backbone to produce the segmentation masks. The dataset, consisting of 7076 256×256px patches, was heavily imbalanced: the dominant class – stabilized roads – represented 82.6 % of all roads, which severely affected both the performance and the applicability of the model. To cope with this, oversampling and undersampling, implemented by weighting, was used. As a result, the accuracy was very optimistic (98.73), while the other metrics showed worse performance – IoU of 41.99 and F1 of 46.61. The model suffered the most from class imbalance and ground-truth label imprecision.

Garcia-Molsosa et al. have successfully used a UNet-based CNN to extract archaeological features from historical maps. All detectors have been able to detect at least 90 % of the features [17].

Many recent works involve the use of convolutional neural networks (CNNs) [7]. One of the popular ones is the UNet architecture [6, 15] and its variations [16]. In general, good results have been achieved on homogenous datasets, however these models often lack flexibility and perform poorly when met with unfamiliar features [12].

Other works have combined CNNs and other methods to maximize the strengths of both. In 2021, Chen et al.[18] combined CNNs (BDCN) and watershed techniques to extract closed shapes on a 1925 atlas of Paris. Including watersheding raised more than doubled F1 score as opposed to a pure-CNN approach.

Other approaches have, however, been also explored. Classical computer vision methods, such as *k*-means clustering, line extraction, image filtering, region growing, shape descriptors etc., have been implemented with various degree of success[19]. On the experimental side is the evolutionary approach, as presented in [20].

1.4 This work’s approach

In this work, two different models will be implemented: UNet and TransUNet.

UNet is a CNN, first developed for biomedical image segmentation in 2015. Today, UNet and its derivations, such as UNet++, ResUNet and others, is the gold standard for diverse segmentation tasks. Because it is so well-established, it was chosen to be the first model in this thesis.

TransUNet, on the other hand, is an architecture that was not researched much in the context of the segmentation of historical maps. It is a hybrid Transformer-CNN model proposed in 2021. It was chosen as a contrast to the well-established UNet and with the intent to compare the results of the two deep learning architectures.

1.5 Definitions

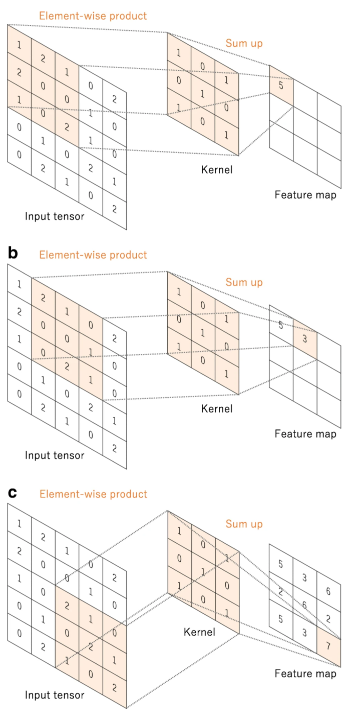


Figure 1.1 Visualization of convolution in a CNN. Source: [22]

1.5.1 Convolutional neural networks

The convolutional neural network (CNN) is a type of an artificial neural network that uses the mathematical operation of convolution at its core. It is specialized for processing grid-like data, such as images. CNNs have multiple types of layers. Here, I will describe the most important ones. Note that I will present CNN in the context of image processing [7, 21].

1.5.1.1 Convolutional layer

The convolutional layer is the building block of the CNN. Its purpose is to extract features from the input image.

Mathematically, convolution is an operation on two real-valued functions f, g. Typically, it is denoted with an asterisk: $f * g$. In all generality, it is defined as the integral of the product of the two functions after one is reflected about the y-axis and shifted. In the context of two-dimensional images, it is practical to work with the discrete definition of convolution instead:

$$S(i,j) = (I*K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n) \tag{1.1}$$

In this notation, I is the input image and K is the kernel (filter). In case of multi-dimensional input (e.g., color channels), the kernel extends through the full depth of the input. During the forward pass, the filter convolves (”slides”) across the input image and the dot product between each of position of the input and the entries of the filter is calculated. The output matrix is referred to as the feature or the activation map. Typically, multiple activation maps are produced at the same time, which are stacked along the depth dimension.

The kernel is typically much smaller than the input, which significantly reduces the number of parameters, compared to traditional neural networks. This is referred to as **sparse connectivity** – instead of all pixels, output is only dependent on those included in the kernel (this is called

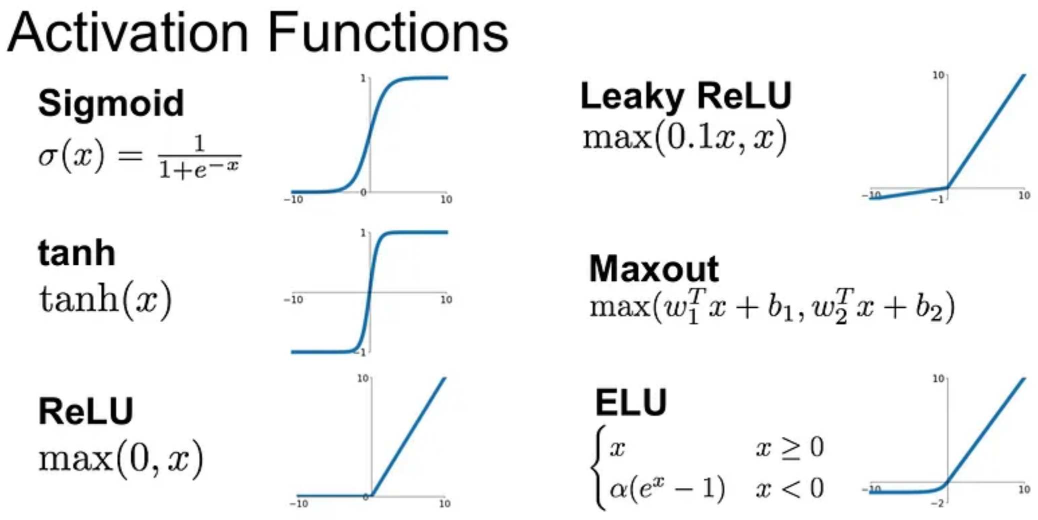


Figure 1.2 Some of the activation functions commonly used in CNNs. Source: https://medium.com/dataseries/basic-overview-of-convolutional-neural-network-cnn-4fcc7dbb4f17

its receptive field). This reduces the model's memory requirements and, consequently, improves computational efficiency.

Another quality of CNNs is **parameter sharing**. In traditional NNs, an individual parameter is kept for each pair of neurons. In contrary, in CNNs the kernel parameters are shared across all positions of the input. Therefore only one set of parameters has to be learnt. The sharing also means that the layer is equivariant to translation (shift invariance).

To further reduce the computational cost, some kernel positions can be skipped. **Stride** s refers to sampling only s pixels in each direction in the input. Mathematically, strided convolution ($s > 1$) is equivalent to convolution, then downsampling.

To prevent the image from shrinking at each layer, zero padding is usually used.

1.5.1.2 Activation function

The convolution is often followed by an activation function to introduce non-linearity to the model. This allows the model to learn and perform more complex tasks.

The most widely used activation function is the Rectified Linear Unit (**ReLU**). It is linear for positive values and zero for negative values:

$$y = \max(0, x) \tag{1.2}$$

Its great advantages are its computational efficiency and fast convergence, compared to other functions. It is nearly linear so, like linear models, it is easy to optimize and generalizes well. However, it also has some disadvantages: it is not zero-centered, and the gradient at the negative region is zero. With the gradient equal to zero, none of the weights will be updated during backpropagation. This is known as the ”dying ReLU problem”. Once a neuron gets negative, it is unlikely for it to recover. Such ”dead” neurons become inactive as they do not take part in discriminating inputs and are unable to learn any further. High learning rate and large negative bias negatively contribute to this.

Leaky ReLU tackles this problem. Unlike ReLU, it has a small slope for negative values:

$$y = \max(0.01x, x) \tag{1.3}$$

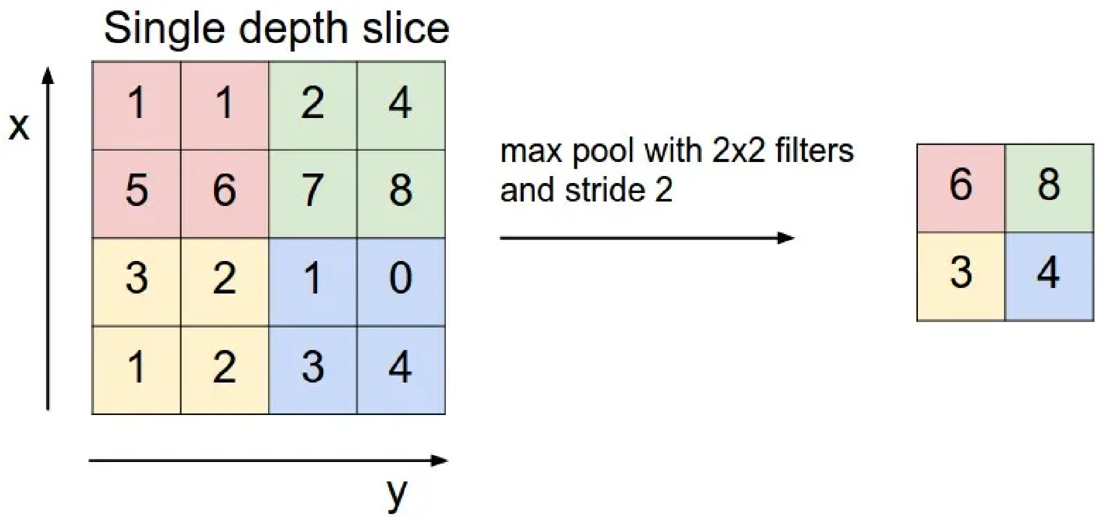


Figure 1.3 Example of maxpool. Source: https://medium.com/dataseries/basic-overview-of-convolutional-neural-network-cnn-4fcc7dbb4f17

This fixes the ”dying ReLU problem” as it doesn't have zero-slope parts. The generalization of Leaky ReLU is the Parametric ReLU (**PReLU**) where the slope is parameterized:

$$y = \max(ax, x) \tag{1.4}$$

Softmax is a special type of activation function that is often at the very end of a CNN architecture. It is used to normalize the output of a network to a probability distribution over n predicted output classes. A vector X's softmax is calculated for each of its output class i as follows:

$$y_i = \frac{e^{y_i}}{\sum_{j=1}^{n} e^{y_j}} \tag{1.5}$$

Some of the other less commonly used activation functions include the following:

- sigmoid: $y = \frac{1}{1+e^{-x}}$,
- tanh: $y = \tanh(x)$,
- ELU: $\begin{cases} x & x \geq 0 \\ a(e^x - 1) & x < 0 \end{cases}$.

1.5.1.3 Pooling layer

A pooling layer produces summary statistic of groups of pixels. Its purpose is to reduce the number of parameters and computation in the network, controlling overfitting by progressively reducing the spatial size of the network. It is designed to make the input more resistant to small translations. It is also useful for unifying inputs of varying size. The depth dimension remains unchanged. Popular pooling functions include max, average, weighted average and the L^2 norm.

1.5.1.4 Other layers

In a **fully connected layer**, the neurons have a complete connection to all the activations from the previous layers. They are usually found at the final phase of a CNN.

Batch normalization is an algorithmic method that is designed to make the training of neural Networks faster and more stable. It consists of normalizing activation vectors from hidden layers using mean and variance of the current batch:

$$X_{norm} = \frac{X - \mu}{\sqrt{\sigma^2 + \epsilon}} \tag{1.6}$$

It then calculates the layer's output by applying a linear transformation with γ and β, two trainable parameters allowing to adjust the standard deviation and the bias:

$$\hat{X} = \gamma * X_{norm} + \beta \tag{1.7}$$

This normalization step is applied right before or right after the nonlinear function [23, 24].

A **dropout layer** is a regularization technique that can mitigate overfitting. This refers to dropping out nodes in the network (with a probability of p). All forward and backward connections of these "dropped" nodes are temporarily removed, meant to prevent overfixation on certain nodes and improving the model's generalization ability.

Transpose convolution, also known as up-convolution or deconvolution, reverses the parameters of the original convolution kernel upside down and flipped horizontally, and fills the spaces between and around the elements of the original image.

1.5.2 UNet

UNet[8] is a well-established encoder-decoder convolutional neural network, popular for image segmentation. It was first developed for biomedical image segmenation, but has since found uses in many visual recognition tasks, such as semantic segmentation, instance segmentation, and even in tasks like image-to-image translation and image synthesis.

The architecture is U-shaped, giving it its name, with four encoder and four decoder blocks connected via a "bridge". Each encoder block consists of the following:

- Two 3x3 convolutions followed by the ReLU activation function. The convolution does not use any padding – in effect, each convolution reduces the resolution by two pixels in the height and width dimensions.
- 2x2 maxpooling operation with stride 2 – this effectively reduces the dimensions by half.

This encoder block represents a downsampling step. Each step doubles the number of feature channels.

The decoder block, representing an upsampling step, includes the following:

- A 2x2 upsampling (transpose) convolution – doubling the dimensions.*
- Concatenation with the corresponding cropped activation map from the contracting path. The cropping is necessary due to the loss of border pixels in every convolution.
- Two 3x3 convolutions, followed by ReLU.

Each upsampling step halves the number of feature channels. At the final layer a 1x1 convolution is used to map each 64-component feature vector to the desired number of classes. In total the network has 23 convolutional layers. In the original paper, stochastic gradient descent is used for training. The authors also include dropout layers at the end of the contracting path to regularize the network and prevent overfitting by randomly dropping out units during training, thereby reducing the reliance of the model on specific activations and improving its generalization ability.

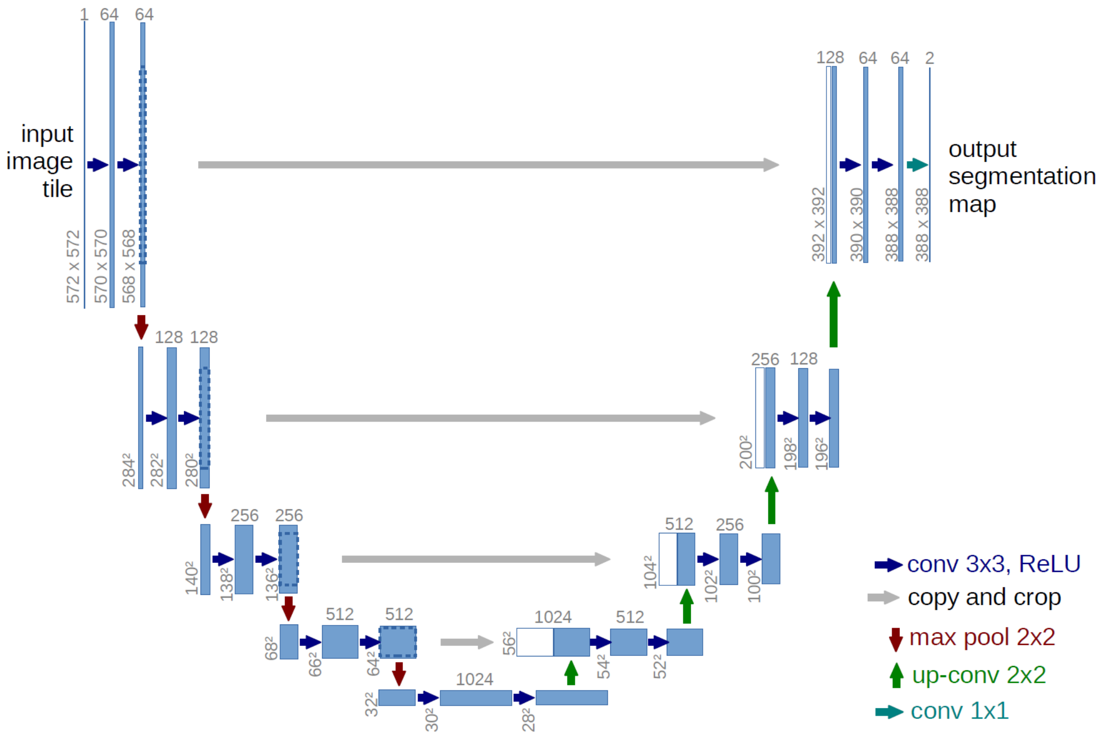


Figure 1.4 UNet architecture. Source: [8]

1.5.3 Transformers

A vision transformer (ViT) [9] is a deep learning architecture for computer vision based on the multi-head attention mechanism designed. It is closely derived from the original transformer that was made for use in natural language processing[25]. Similarly to how text is broken into tokens in transformers, ViTs work with images split into patches.

ViTs lack some of the properties CNNs have, such as translation invariance and locality, and therefore do not generalize well on small to mid-sized datasets. On large enough datasets, however, ViTs yield results competitive with those of CNNs. They can therefore be pretrained on large public datasets, such as ImageNet-21k, and then transferred to tasks with limited training data and fine-tuned.

An input image is transformed into a sequence of fixed-size flattened patches and linearly embedded using a mapping with trainable linear projection. Alternatively, the input sequence can be formed from feature maps of a CNN, creating a hybrid architecture. The sequence of embedded patches is then fed into a transformer encoder, which consists of multiple layers of self-attention mechanisms and feedforward neural networks. Within each layer, self-attention allows the model to capture global dependencies between different parts of the input sequence, while the feedforward neural networks provide non-linear transformations to the attention outputs. Additionally, residual connections and layer normalization are applied to facilitate gradient flow and stabilize training. After passing through the transformer encoder layers, the final sequence representation is typically processed by a classifier head to make predictions for the task at hand, such as image classification or object detection.

Conversely to a CNN, In a ViT, the self attention layers are global and only the MLP layers are local and shift invariant. All spatial relations between the patches need to be learned from scratch.

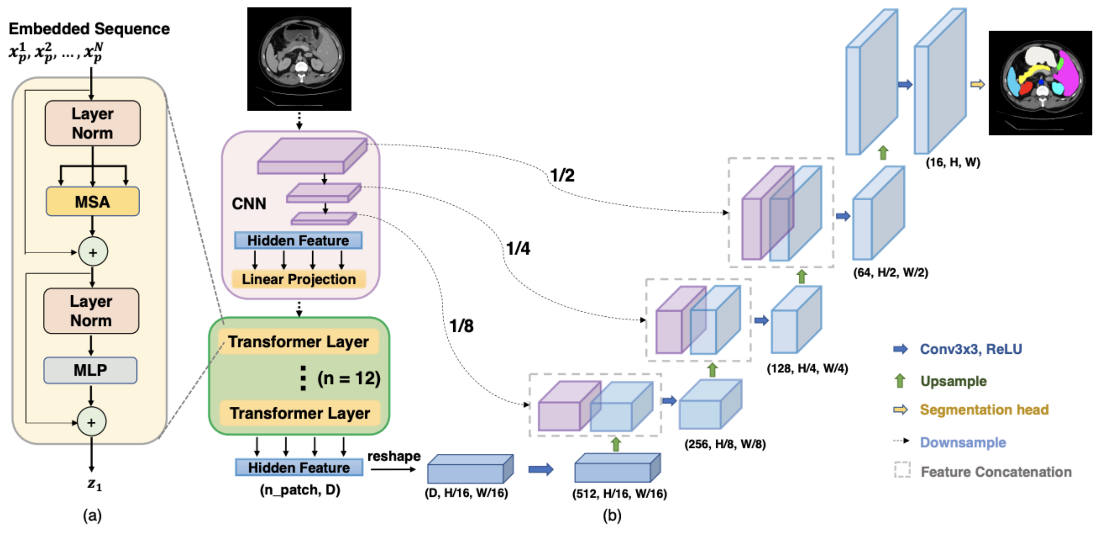


Figure 1.5 TransUNet architecture. Source: [11]

1.5.4 TransUNet

TransUNet[11] is the first segmentation model built from the Transformer. It was first proposed for medical image segmentation. Due to the locality of convolution operations, CNNs can effectively exploit only short-range information. Transformers, on the other hand, perform well at long-range dependency but are limited in terms of exploring local features. TransUNet combines these two approaches to maximize the efficiency of segmentation. In a nutshell, the paper argues that a hybrid CNN-Transformer encoder yields better results because the CNN helps exploit local features better.

The authors also claim that a straightforward usage, where the transformer is directly applied to the entire input image, might not fully capture local spatial information crucial for accurate segmentation. Therefore, the authors propose an architecture where the Transformer is applied to local patches of the input image, enabling it to capture both long-range dependencies and local details effectively. This approach allows TransUNet to achieve state-of-the-art performance in medical image segmentation tasks by leveraging the strengths of both CNNs and Transformers.

The CNN is first used as a feature extractor to generate a feature map for the input. UNet-like skip connections are present for extracting low-level spatial information.

1.5.5 Transfer learning

Transfer learning is a machine learning technique where a model trained on one task is deployed to perform a related task. By transferring knowledge from the pre-trained model to the target task, transfer learning often leads to improved performance and faster convergence, particularly in scenarios with limited training data. Pre-training can be done ad hoc but it is more common to use models trained on large public datasets, such as ImageNet.[26, 27]

1.5.6 Data augmentation

Data augmentation is a technique that is used to reduce overfitting during training a model. Simple alterations, such as vertical and horizontal flipping, cropping, zooming in, adding noise, changing contrast etc., randomly applied to training images. This artificially enlarges and diversifies the dataset, which is especially useful when the available data is limited [28]. In effect, data augmentation improves the generalization of a classifier.[7]

1.5.7 Metrics of evaluation

Four metrics have been chosen to assess the performance of the models in this work: Intersection over Union, precision, recall and F1 score.

All presented metrics are based on the computation of a confusion matrix for a binary segmentation mask, which contains the number of true positive (TP), false positive (FP), true negative (TN), and false negative (FN) predictions. The value ranges of all presented metrics span from zero (worst) to one (best).

Intersection over Union (often shortened to IoU, also called the Jaccard index) is a metric frequently used in computer vision tasks that estimates how well a predicted mask or a bounding box matches the ground truth data. It is calculated by dividing the overlap between the predicted and ground truth annotation by the union of these.

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|} = \frac{TP}{TP + FP + FN} \tag{1.8}$$

Precision and *recall* are both directly from the confusion matrix. Precision is the ratio of true positive predictions to the total number of positive predictions, while recall is the ratio of the true positives to the sum of true positives and false negatives.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{1.9}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{1.10}$$

F1 score (also called the Dice Coefficient) is defined as harmonic mean of precision and recall.

$$F_1(A,B) = \frac{|A \cap B|}{|A| + |B|} = \frac{2TP}{2TP + FP + FN} \tag{1.11}$$

Chapter 2

Practical part

The main focus of this chapter is the analysis of the historical map dataset. I will follow by detailed explanation of the chosen implementation, and of experiments I will be carrying out in the following chapter.

2.1 Dataset

The dataset [29] consists of two subsets.

The first is the Paris dataset. It is derived from a larger corpus numbering over 1500 maps obtained from the collections of the Bibliothèque nationale de France (French National Library) and the Bibliothèque historique de la Ville de Paris (Historical Library of the City of Paris). They cover a period from 1760 to 1994, most of them being from the years 1800–1950. The dataset I will be dealing with is made up of 330 1000×1000px patches randomly cut from the maps of the original corpus, divided into 300 training samples and 30 evaluation samples.

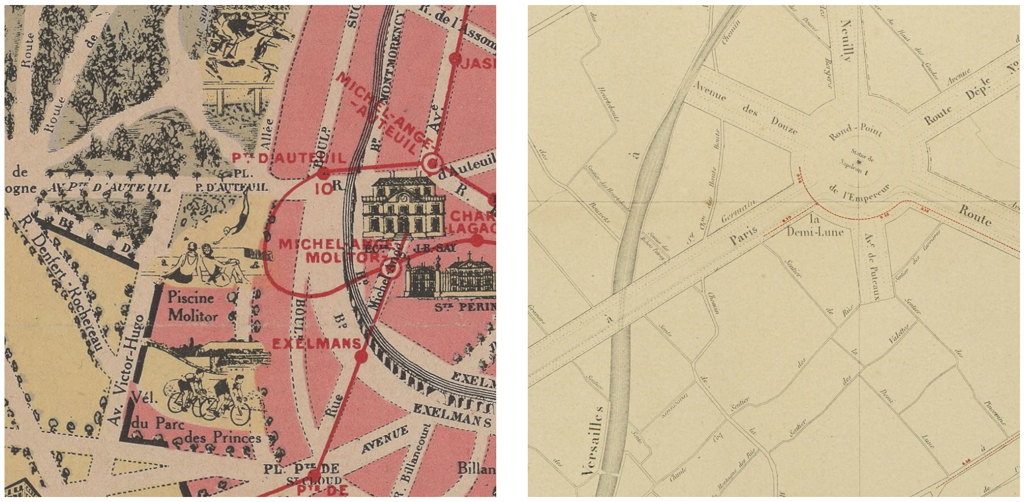
The other is the World dataset, derived from the World corpus. The authors aimed to obtain a balanced sample of urban mapping at global level. It consists of 256 maps representing cities from all of the world, including Asia, Middle East, Africa, Latin America, Oceania, North American and Europe. The corpus was carefully constructed from a number of collections and databases, such as Bibliothèque nationale de France, Library of Congress, Harvard University, David Rumsey Historical Map Collection and other. All 256 maps were used for the training set, one random patch from each map, and another 49 patches from randomly chosen maps make up the evaluation set.

The datasets were then manually annotated. They were designed for the semantic segmentation either into 3 or 5 classes (2+1 or 4+1 ontology, respectively). The 4+1 ontology the practical part will be focusing on comprises these five classes:

- frame (black),
- water (blue),
- blocks (magenta),
- non-built (cyan),
- road network (white). [1]

[1]In the 2+1 ontology, the *water*, *blocks* and *non-built* classes are grouped in a *map content* class.

The Paris dataset, while culturally and geographically homogeneous, suffers from information cluttering and overlay (2.1a). In many cases, the maps contain overlapping information of various nature: the roads, the metro, the water system, the catacombs, the administrative divisions etc. This is because city maps were used as a tool for urban planning until the late 19th century. Some pictures also suffer from low contrast (2.1b).



(a) (b)

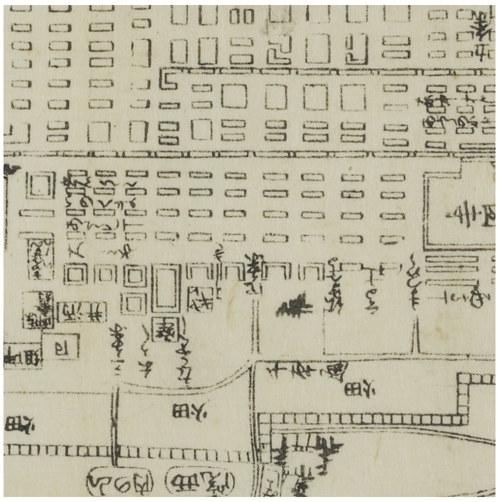
Figure 2.1 Examples of the challenges detected in the Paris dataset: (a) visual cluttering and overlay, (b) low contrast.

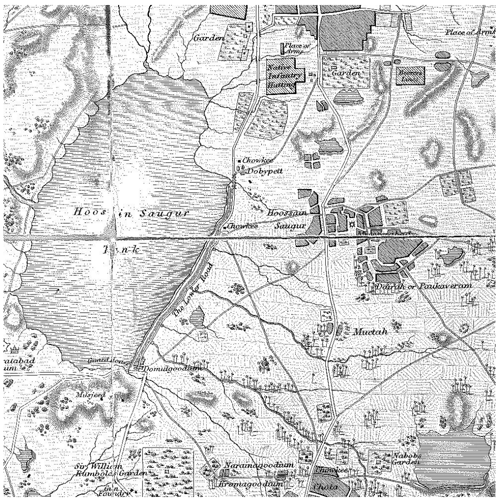
The World dataset, on the other hand, is much more heterogeneous. Due to its nature, it is not only the extreme geographic and cultural diversity that must be taken into consideration (2.2a), but also the differences in scale (2.2b), notation, script and urban development, as well as poor quality and scanning imperfections (2.2c).

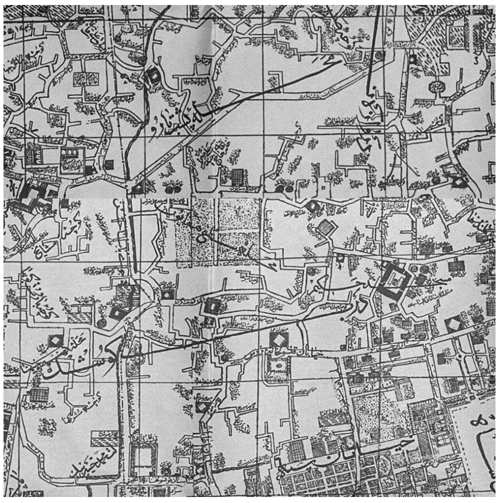
Of the total 635 annotated image patches, 566 make up the training set, 39 make up the validation set and 30 make up the test set. The training set has the following distribution of the target classes:

Dataset	Frame	Water	Blocks	Non-built	Road network
Paris	0.2358	0.0262	0.3381	0.1919	0.2079
World	0.2219	0.1212	0.1749	0.3594	0.1226

Table 2.1 Class distribution of the training data of the two datasets.


(a)


(b)


(c)

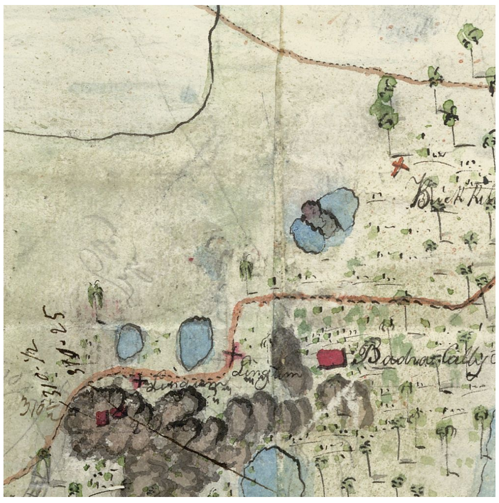

(d)

Figure 2.2 Examples of the challenges detected in the World dataset: (a) cultural differences in cartography, (b) inconsistent scale and source imperfections, (c) differences in urban development, (d) author's artistic style.

From the distribution, we can see that the two datasets differ significantly. The Paris dataset has a much higher ratio of blocks than the rest of the world. This is not a surprising finding, given that Paris was and is one of the densest cities in the world[30]. In the World dataset, on the other hand, the non-built and water classes are more common. Overall, the classes are mostly balanced, except for the water class in the Paris dataset, which is heavily underrepresented.

2.2 Implementation

All models presented here have been trained on Nvidia Tesla V100 Tensor Core GPU, using the shared resources of FIT's Datalab, with the kind permission of the team.

The programming language of the code is Python, which is the de-facto choice for machine learning related work, thanks to its wide support of data science libraries. Here, I will shortly introduce some of the major libraries used in the code.

PyTorch is a machine learning framework, used in the fields of deep learning, computer vision and natural language processing. It is most notable for its tensor computing with the support of GPU acceleration and deep learning tools. Beside PyTorch, I am also using other libraries closely related to PyTorch, such as Torchvision and Torchmetrics.

NumPy offers tools for array operations, as well as many high-level mathematical functions from such domains as linear algebra and random number generation.

Matplotlib is a plotting library, useful for making graphs and other visualizations.

The results are then presented in a *Jupyter Notebook*, an interactive computational environment, which supports Python, among many other programming languages.

2.3 Training

For the elementary models, architecture from the original paper is used, with added batch normalization in UNet. The following settings were used for training:

- batch size: 4,
- number of epochs: 200,
- learning rate: 0.001,
- loss function: cross entropy,
- optimizer: Adam,
- number of heads: 4 (TransUNet only).

The 1000×1000px mages are padded so that their dimensions are 1024×1024, which is needed for a smooth pass through the architectures. Afterwards they are normalized. No postprocessing is applied. The models will be evaluated on the validation set.

2.4 Experiments

Besides training the elementary UNet and TransUNet models as described above, I will carry out four experiments to determine the best hyperparameters for the final segmentation models.

2.4.1 Experiment 1: Transfer learning

This experiment will observe the effects of transfer learning on the models. For UNet, a ResNet50 encoder pretrained on ImageNet will be used; for TransUNet, the vision transformer will be pretrained on ImageNet21k.

2.4.2 Experiment 2: Data augmentation

In this experiment, I will train a UNet and a TransUNet model using data augmentation. I will then compare these two models with the elementary ones and watch the effects of data augmentation on learning of the model and its performance. The following modifications will be applied:

- vertical flip,
- horizontal flip,
- center crop and resize,
- random crop and resize,
- random erase,
- random color jitter,

2.4.3 Experiment 3: Data augmentation + learning rate

Learning rate, also called the step size, is a hyperparameter that defines the adjustment in the weights of a network while moving toward a minimum of a loss function [31]. This experiment will observe the effects of increasing/decreasing the hyperparameter: the models will be trained with learning rate of 0.01 and 0.0001, respectively (as opposed to the learning rate of 0.001 used in the elementary model).

2.4.4 Experiment 4: Data augmentation+ + Leaky ReLu

In the original paper, ReLU is used as the activation function for UNet[8]. The aim of this experiment is to find whether using Leaky ReLU solves the Dying ReLU problem and whether it has a positive effect on the overall performance of the model.

Chapter 3

Results

In this chapter, I will present the results of the experiments laid out in the previous chapter. Discussion of the results follows.

3.1 Elementary models

As described in detail in the previous chapter 2 Practical part, two segmentation models, UNet and TransUNet, were trained. In the following figure, training and validation loss of the models are presented.

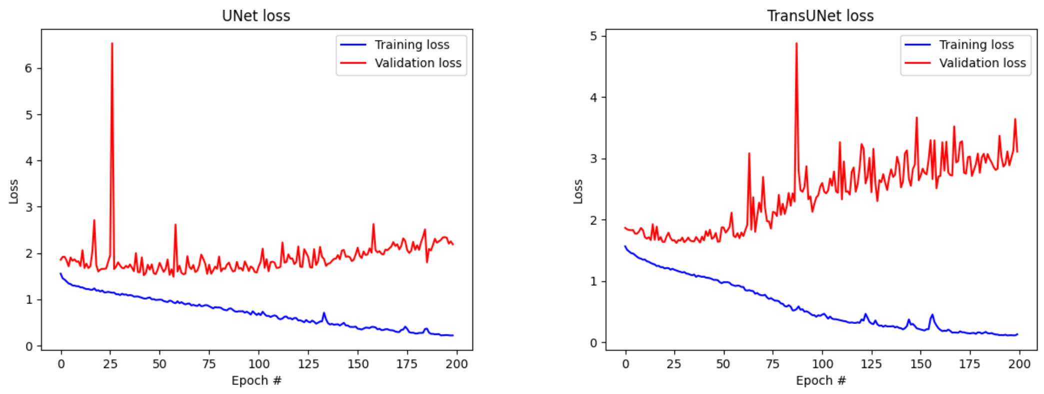


Figure 3.1 Training and validation loss of the elementary models.

Inspecting the graphs of training and validation loss, we can roughly estimate the epoch when the model starts to overfit. Overfitting generally occurs when training loss is still decreasing, while validation loss is stagnating or starts to increase again. In the case of UNet, this seems to be somewhere around epoch 75, and even sooner for TransUNet, around epoch 50. I will therefore choose models at these epochs, which is a sign of overfitting. For further analysis, I will choose models at these epochs, at which I suspect they are not overtrained yet. I will demonstrate the models' outputs on four random samples from the validation set and compare the model's predictions to the ground truth.

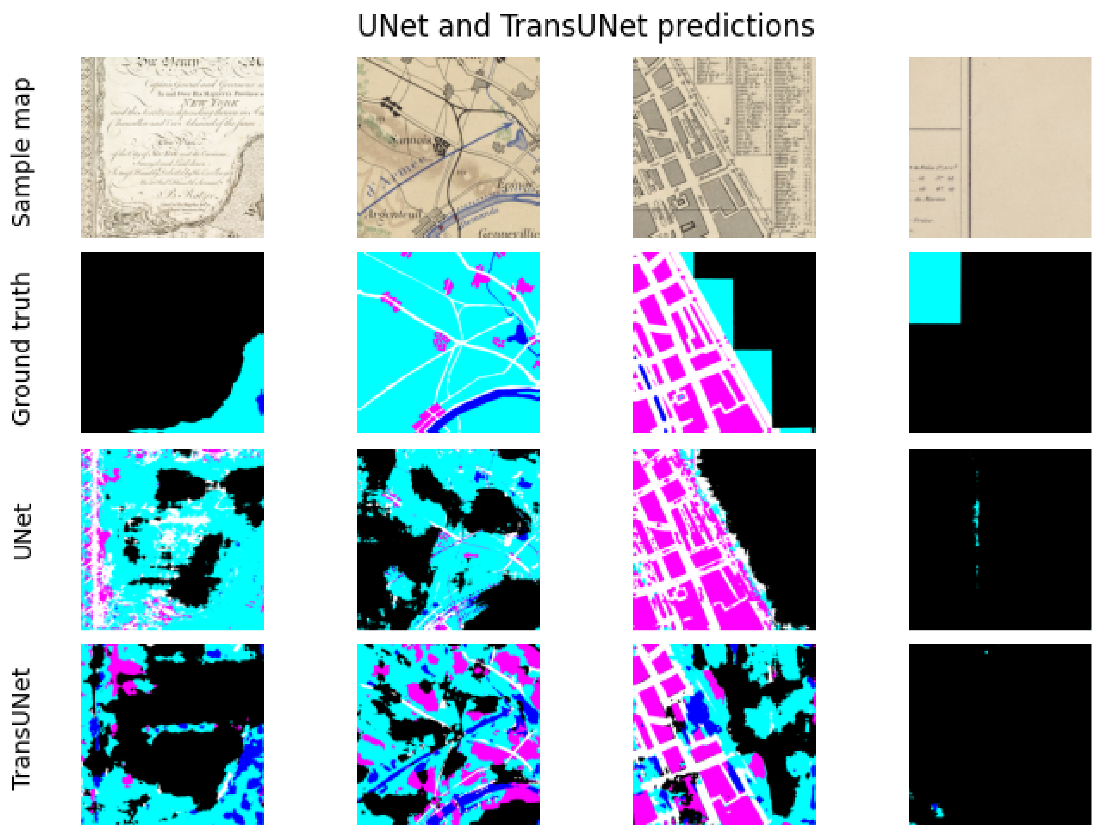


Figure 3.2 UNet (epoch=75) and TransUNet (epoch=50) predictions.

Judging from these samples, we can see that neither of the models perform well on the validation data. For the first sample, both models mistake the text that is a part of frame as map content, and try to segment it as if it were so. The predictions for the second sample are also wrong and uninterpretable. In the third sample, both models are able to make out the basic street layout but there is a lot of noise and artifacts in the TransUnet prediction. Both models successfully detect non-map content in the fourth sample. They do not recognize the non-built area in the upper left corner, although without further context, it is almost impossible to deduce that it is not a part of the frame.

In brief, the elementary models yield poor results. This can also be seen in the mean evaluation metrics:

Metric	UNet	TransUNet
IoU	0.1866	0.1987
Precision	0.3450	0.3687
Recall	0.2879	0.2937
F1	0.2578	0.2716

Table 3.1 Overview of mean evaluation metrics of the elementary models.

Although TransUNet is slightly better in all of the metrics, overall, the results are unsatisfactory. The following four experiments will seek to improve the performance of the models.

3.2 Experiment 1: Transfer learning

In this experiment, I used models that had been pretrained on ImageNet and ImageNet21k, respectively.

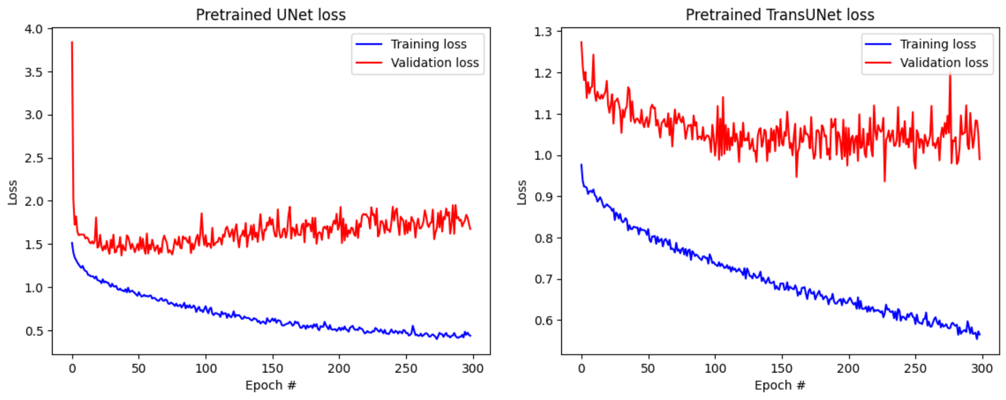


Figure 3.3 Experiment 1: Training and validation loss of the two models.

Based on the graphs of the loss functions, I decide to stop the training at epoch 40 for UNet and at epoch 100 for TransUNet.

Metric	UNet	Pretrained UNet	TransUNet	Pretrained TransUNet
IoU	0.1866	0.2460	0.1987	0.2327
Precision	0.3450	0.3935	0.3687	0.3931
Recall	0.2879	0.3648	0.2937	0.3802
F1	0.2578	0.3169	0.2716	0.3202

Table 3.2 Experiment 1: Overview of mean evaluation metrics of the elementary models and the pretrained models.

Comparing the metrics, it is evident that the pretrained models perform significantly better than the elementary ones. Therefore, pretraining will be included in the final model.

3.3 Experiment 2: Data augmentation

For this experiment, I have trained models that use data augmentation for each of the two discussed architectures.

Data augmentation seems to have slowed down the overfitting for UNet. I choose stopping epoch of 150 for UNet and 50 for TransUnet. Here are the result metrics:

Metric	UNet	UNet w/ data augmentation	TransUNet	TransUnet w/ data augmentation
IoU	0.1866	0.2468	0.1987	0.2448
Precision	0.3450	0.3931	0.3687	0.3824
Recall	0.2879	0.3545	0.2937	0.3602
F1	0.2578	0.3210	0.2716	0.3212

Table 3.3 Experiment 2: Overview of mean evaluation metrics of the elementary models and models trained on augmented data.

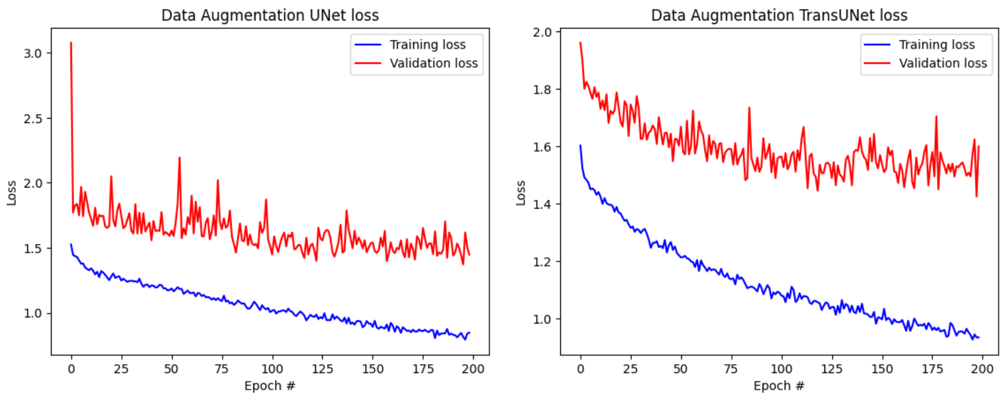


Figure 3.4 Experiment 2: Training and validation loss of the two models.

We can see that training the models on an augmented dataset has a positive impact on its performance. This is again a feature that will be kept in the final model.

3.4 Experiment 3: Data augmentation + learning rate

This experiment examines TransUNet trained on augmented data with learning rates of 0.01 and 0.0001 and compares to the results to the. Analogically to the experiments before, I deduce the epochs at with to stop training from plots of the training loss. The resulting metrics are as follow:

Metric	TransUNet(LR=0.0001)	TransUNet(LR=0.001)	TransUNet(LR=0.01)
IoU	0.2189	0.2448	0.2488
Precision	0.3509	0.3824	0.3883
Recall	0.3381	0.3602	0.3552
F1	0.2957	0.3212	0.3284

Table 3.4 Experiment 3: Overview of performance of TransUNet models with different learning rates.

Increasing learning rate to 0.01 has virtually no effect on the overall performance; decreasing learning rate to 0.0001 has actually even worsened it. Therefore, the original learning rate will be kept for the final model. Because of this result, the experiment was not replicated on UNet.

3.5 Experiment 4: Data augmentation + Leaky ReLu

In this last experiment, I train UNet with Leaky ReLU as its activation function. The stopping epoch is again deduced from the loss graph. Comparing with the elementary UNet model, we get the following results:

It is apparent from the results that using Leaky ReLU as an activation function has very little effect on the overall performance of the model. Because the effect is negligible, the experiment was not repeated with the TransUNet model.

3.6 Final model

The results of the previous experiments can be summed up like so:

Metric	UNet w/ data augmentation	Leaky ReLU UNet w/ data augmentation
IoU	0.2468	0.2462
Precision	0.3931	0.3893
Recall	0.3545	0.3560
F1	0.3210	0.3184

Table 3.5 Experiment 4: Overview of mean evaluation metrics of the data augmented UNet and data augmeneted UNet with Leaky ReLU as its activation function.

- Experiment 1: Transfer learning has a positive impact on both models.
- Experiment 2: Data augmentation has a positive impact on both models.
- Experiment 3: Decreasing or increasing the learning rate has no positive impact on the models.
- Experiment 4: Switching ReLU for Leaky ReLU has a negligible impact on the models.

Based on these findings, I can construct the final two models. They differ from the elementary models in that they use both transfer learning (pretraining) and data augmentation. Training these two models results in the following loss plots:

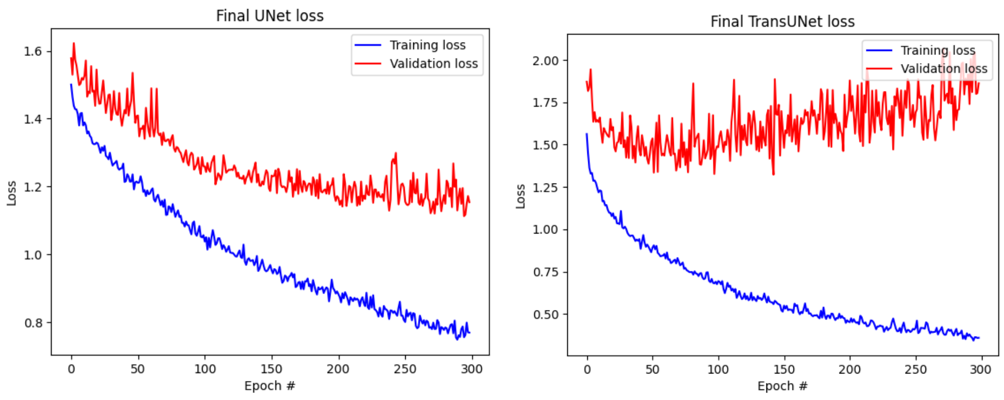


Figure 3.5 Training and validation loss of the final models.

Analogically to before, the stopping epoch can be estimated from the graphs – here, 200 for UNet and 75 for TransUNet. As these are the final models, we can finally use the holdout test set to evaluate the performance. As with the elementary models, the predictions will be illustrated on four random samples from the test set, together with a detailed class-wise overview metrics.

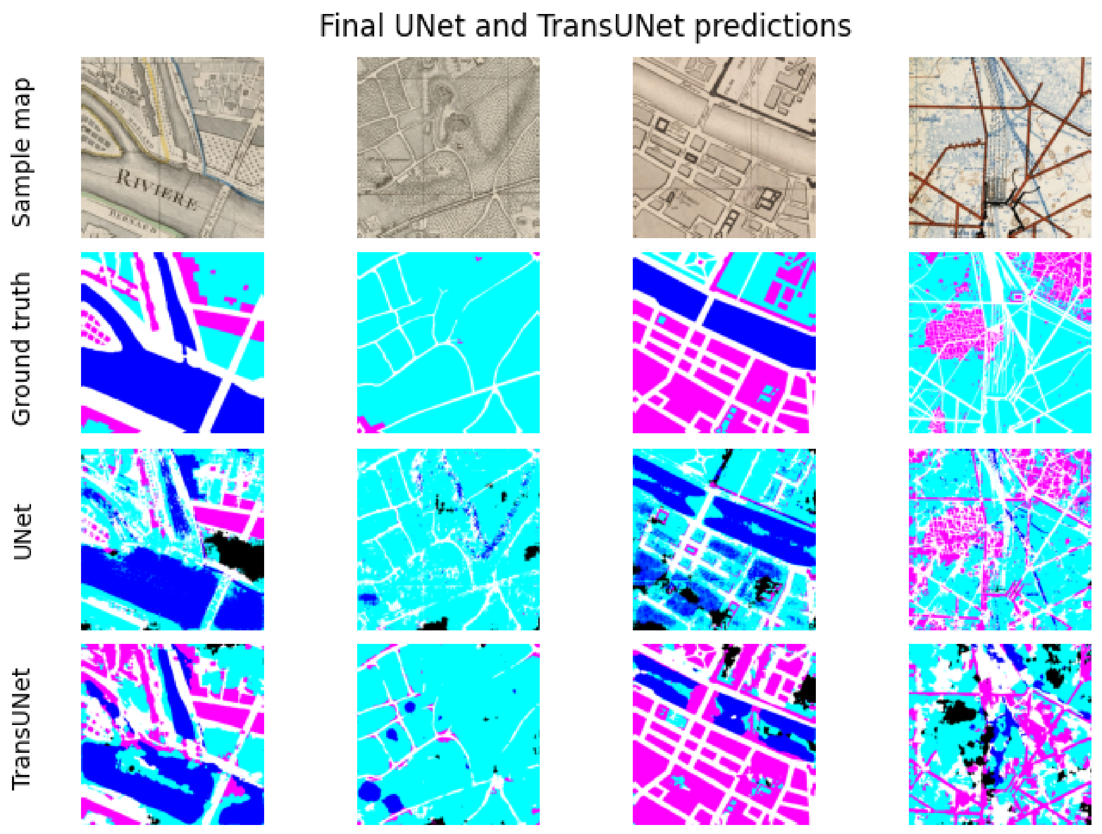


Figure 3.6 Final UNet (epoch=200) and TransUNet (epoch=75) predictions.

3.7 Discussion

Visually, the final models behave better than the elementary models at the beginning. They still show big inaccuracies but they can generally detect the rough layout of the streets roads. This correlates with the metrics that have gone up roughly by 10 %, compared to the elementary models.

Having a look at the metrics per class, it is apparent that the water class has the worst results for all of the metrics. This is most probably due to the fact that it is the least represented class in the dataset. Road network, on the other hand, receives good score on IoU, recall and F1.

If we compare the two models, their results are comparable, although TransUNet has slightly better scores. It is therefore safe to say that, for the task of semantic segmentation of historical maps, TransUNet is a suitable alternative to the well-known UNet.

The results I achieved are by no means the best they could be. R. Pétitpierre[15] achieved mean IoU of 0.6363 on the Paris dataset and 0.5595 on the World dataset. I see several ways that this work's results could be improved.

First, careful preprocessing and postprocessing of images could be beneficial. It was shown in the state-of-the-art section that combination of deep learning methods and traditional approaches can have a very good impact on the performance of the model[13, 18]. For preprocessing, I suggest to first detect the non-map content of the images before the actual segmentation. For postprocessing, transformation such as line detection might help with creating sharp edges between the classes, instead of patches of different classes.

Metric	UNet	TransUNet
IoU	0.2759	0.2839
Frame	0.2845	0.2941
Water	0.0823	0.0855
Block	0.3528	0.3516
Non-built	0.2983	0.3104
Road network	0.3617	0.3777
Precision	0.4278	0.4327
Frame	0.3439	0.3532
Water	0.2999	0.2856
Block	0.4539	0.4759
Non-built	0.5823	0.5912
Road network	0.4592	0.4575
Recall	0.4136	0.4236
Frame	0.3528	0.3616
Water	0.1038	0.1154
Block	0.6143	0.6249
Non-built	0.3355	0.3440
Road network	0.6616	0.6723
F1	0.3450	0.3411
Frame	0.3193	0.3042
Water	0.0994	0.1034
Block	0.4124	0.4242
Non-built	0.3911	0.3611
Road network	0.5030	0.5128

Table 3.6 Overview of performance of final models

Second, more hyperparameter tuning and experiments could bring an improvement in the performance. One might also experiment with batch size, number of heads and different types of data augmentation, optimizers and activation functions. There is a massive number of hyper-parameters that can be fine-tuned but for which there is unfortunately not enough space in this work.

Lastly, I suggest pretraining on a more specialized dataset. While pretraining on ImageNet did help, it is worth exploring what the effects of other sources bases could be. R. Pétitpierre[15], for example pretrained the Paris dataset on the World dataset and vise versa.

Conclusion

At the beginning of this work, I set four major objectives to achieve. In the theoretical part, I introduced the relevant theory and described current approaches to the task of image segmentation of historical maps. Then, I trained two elementary models, UNet and TransUnet. Next, I carried out four experiments that helped me find the ideal hyperparameters for the final models. Lastly, I showed the results of the final models, compared them, discussed their performance and gave suggestions as to what could be done to improve the results further.

Analysis of historical maps is a difficult task and this work was not an exception. I showed that for this kind of task, TransUNet is a suitable alternative to the well-established UNet. I also showed that data augmentation and transfer learning can have positive impact on the models. It is my wish that this work gives an incentive for further study and development in the tough task of analysis of historical documents and maps.

Bibliography

1. JELÍNKOVÁ, Kateřina. *Česká kartografie*. Petrklíč, 2016. ISBN 978-80-7229-593-7.
2. ROBINSON, Arthur Howard et al. *Elements of cartography*. 6th ed. Wiley New York, 1995. ISBN 0471555797.
3. SHAPIRO, L.G.; STOCKMAN, G.C. *Computer Vision*. Prentice Hall, 2001. ISBN 9780130307965. Available also from: `https://books.google.cz/books?id=FftDAQAAIAAJ`.
4. YU, Ying; WANG, Chunping; FU, Qiang; KOU, Renke; HUANG, Fuyu; YANG, Boxiong; YANG, Tingting; GAO, Mingliang. Techniques and Challenges of Image Segmentation: A Review. *Electronics*. 2023, vol. 12, no. 5. ISSN 2079-9292. Available from DOI: `10.3390/electronics12051199`.
5. OTSU, Nobuyuki. A Threshold Selection Method from Gray-Level Histograms. *IEEE Transactions on Systems, Man, and Cybernetics*. 1979, vol. 9, no. 1, pp. 62–66. Available from DOI: `10.1109/TSMC.1979.4310076`.
6. MÄYRÄ, Janne; KIVINEN, Sonja; KESKI-SAARI, Sarita; POIKOLAINEN, Laura; KUMPULA, Timo. Utilizing historical maps in identification of long-term land use and land cover changes. *AMBIO A Journal of the Human Environment*. 2023. Available from DOI: `10.1007/s13280-023-01838-z`.
7. GOODFELLOW, Ian J.; BENGIO, Yoshua; COURVILLE, Aaron. *Deep Learning*. Cambridge, MA, USA: MIT Press, 2016. `http://www.deeplearningbook.org`.
8. RONNEBERGER, Olaf; FISCHER, Philipp; BROX, Thomas. *U-Net: Convolutional Networks for Biomedical Image Segmentation*. 2015. Available from arXiv: `1505.04597 [cs.CV]`.
9. DOSOVITSKIY, Alexey; BEYER, Lucas; KOLESNIKOV, Alexander; WEISSENBORN, Dirk; ZHAI, Xiaohua; UNTERTHINER, Thomas; DEHGHANI, Mostafa; MINDERER, Matthias; HEIGOLD, Georg; GELLY, Sylvain; USZKOREIT, Jakob; HOULSBY, Neil. *An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale*. 2021. Available from arXiv: `2010.11929 [cs.CV]`.
10. OKTAY, Ozan; SCHLEMPER, Jo; FOLGOC, Loïc Le; LEE, Matthew C. H.; HEINRICH, Mattias P.; MISAWA, Kazunari; MORI, Kensaku; MCDONAGH, Steven G.; HAMMERLA, Nils Y.; KAINZ, Bernhard; GLOCKER, Ben; RUECKERT, Daniel. Attention U-Net: Learning Where to Look for the Pancreas. *CoRR*. 2018, vol. abs/1804.03999. Available from arXiv: `1804.03999`.
11. CHEN, Jieneng; LU, Yongyi; YU, Qihang; LUO, Xiangde; ADELI, Ehsan; WANG, Yan; LU, Le; YUILLE, Alan L.; ZHOU, Yuyin. *TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation*. 2021. Available from arXiv: `2102.04306 [cs.CV]`.

12. PETITPIERRE, Rémi; KAPLAN, Frédéric; LENARDO, Isabella di. Generic Semantic Segmentation of Historical Maps. In: *Proceedings of the Workshop on Computational Humanities Research (CHR 2021)*. Amsterdam, NL: CEUR, 2021.

13. CHAZALON, Joseph; CARLINET, Edwin; CHEN, Yizi; PERRET, Julien; DUMÉNIEU, Bertrand; MALLET, Clément; GÉRAUD, Thierry; NGUYEN, Vincent; NGUYEN, Nam; BALOUN, Josef; LENC, Ladislav; KRÁL, Pavel. *ICDAR 2021 Competition on Historical Map Segmentation*. 2021. Available from arXiv: `2105.13265 [cs.CV]`.

14. KAIM, Dominik; KOZAK, Jacek; KOLECKA, Natalia; ZIÓŁKOWSKA, Elżbieta; OSTAFIN, Krzysztof; OSTAPOWICZ, Katarzyna; GIMMI, Urs; MUNTEANU, Catalina; RADELOFF, Volker C. Broad scale forest cover reconstruction from historical topographic maps. *Applied Geography*. 2016, vol. 67, pp. 39–48. ISSN 0143-6228. Available from DOI: `https://doi.org/10.1016/j.apgeog.2015.12.003`.

15. PETITPIERRE, Rémi. Neural networks for semantic segmentation of historical city maps: Cross-cultural performance and the impact of figurative diversity. *CoRR*. 2020, vol. abs/2101.12478. Available from arXiv: `2101.12478`.

16. EKIM, Burak; SERTEL, Elif; KABADAYI, M. Erdem. Automatic Road Extraction from Historical Maps Using Deep Learning Techniques: A Regional Case Study of Turkey in a German World War II Map. *ISPRS International Journal of Geo-Information*. 2021, vol. 10, no. 8. ISSN 2220-9964. Available from DOI: `10.3390/ijgi10080492`.

17. GARCIA-MOLSOSA, Arnau; ORENGO, Hector A.; LAWRENCE, Dan; PHILIP, Graham; HOPPER, Kristen; PETRIE, Cameron A. Potential of deep learning segmentation for the extraction of archaeological features from historical map series. *Archaeological Prospection*. 2021, vol. 28, no. 2, pp. 187–199. Available from DOI: `https://doi.org/10.1002/arp.1807`.

18. CHEN, Yizi; CARLINET, Edwin; CHAZALON, Joseph; MALLET, Clément; DUMÉNIEU, Bertrand; PERRET, Julien. Combining Deep Learning and Mathematical Morphology for Historical Map Segmentation. In: LINDBLAD, Joakim; MALMBERG, Filip; SLADOJE, Nataša (eds.). *Discrete Geometry and Mathematical Morphology*. Cham: Springer International Publishing, 2021, pp. 79–92. ISBN 978-3-030-76657-3. Available from DOI: `10.1007/978-3-030-76657-3_5`.

19. JIAO, Chenjing; HEITZLER, Magnus; HURNI, Lorenz. A survey of road feature extraction methods from raster maps. *Transactions in GIS*. 2021, vol. 25, no. 6, pp. 2734–2763. Available from DOI: `https://doi.org/10.1111/tgis.12812`.

20. HEROLD, Hendrik. An Evolutionary Approach to Adaptive Image Analysis for Retrieving and Long-term Monitoring Historical Land Use from Spatiotemporally Heterogeneous Map Sources. In: 2016. Available also from: `https://nbn-resolving.org/urn:nbn:de:bsz:14-qucosa-199355`.

21. *Course notes in Deep Learning for Computer Vision*. Stanford University, 2023. Available also from: `https://cs231n.github.io/`. Accessed on 10/01/2024.

22. YAMASHITA, Rikiya; NISHIO, Mizuho; DO, Richard Kinh Gian; TOGASHI, Kaori. Convolutional neural networks: an overview and application in radiology. *Insights into Imaging*. 2018, vol. 9, no. 4. Available from DOI: `10.1007/s13244-018-0639-9`.

23. IOFFE, Sergey; SZEGEDY, Christian. Batch Normalization: Accelerating Deep Network Training by Reducing Internal Covariate Shift. *CoRR*. 2015, vol. abs/1502.03167. Available from arXiv: `1502.03167`.

24. *Batch normalization in 3 levels of understanding*. Johann Huber, 2020. Available also from: `https://towardsdatascience.com/batch-normalization-in-3-levels-of-understanding-14c2da90a338#b93c`. Accessed on 16/01/2024.

25. VASWANI, Ashish; SHAZEER, Noam; PARMAR, Niki; USZKOREIT, Jakob; JONES, Llion; GOMEZ, Aidan N.; KAISER, Lukasz; POLOSUKHIN, Illia. *Attention Is All You Need*. 2023. Available from arXiv: `1706.03762 [cs.CL]`.

26. PAN, Sinno Jialin; YANG, Qiang. A Survey on Transfer Learning. *IEEE Transactions on Knowledge and Data Engineering*. 2010, vol. 22, no. 10, pp. 1345–1359. Available from DOI: `10.1109/TKDE.2009.191`.

27. RUDER, Sebastian. *Transfer Learning - Machine Learning's Next Frontier* [`http://ruder.io/transfer-learning/`]. 2017.

28. SHORTEN, Connor; KHOSHGOFTAAR, Taghi M. A survey on Image Data Augmentation for Deep Learning. *Journal of Big Data*. 2019, vol. 6, no. 1. Available from DOI: `10.1186/s40537-019-0197-0`.

29. PETITPIERRE, Rémi. *Historical City Maps Semantic Segmentation Dataset*. Zenodo, 2021. Version 1.0. Available from DOI: `10.5281/zenodo.5513639`.

30. Fewer Parisians, but more Greater Parisians: density in the Île de France. [N.d.]. Available also from: `https://www.lagrandeconversation.com/en/society/fewer-parisians-but-more-greater-parisians-density-in-the-ile-de-france/`. Accessed on 28/12/2023.

31. MURPHY, K.P. *Machine Learning: A Probabilistic Perspective*. MIT Press, 2012. Adaptive Computation and Machine Learning series. ISBN 9780262304320. Available also from: `https://books.google.cz/books?id=RC43AgAAQBAJ`.

Enclosed medium

```
README.md ............................................ read me file
├── thesis.pdf ....................................... this thesis
├── results.ipynb ............. detailed presentation of the results
├── train.py .................. dataset loading and training loop
├── util.py ..................................... auxiliary functions
├── data/img .......................................... source data
│   ├── paris
│   └── world
├── models ............... directory with model implementations
│   ├── unet.py
│   └── transunet.py
└── trained_models ........... trained models and training losses
    ├── unet
    └── transunet
```