CNN-BASED TEMPLATE MATCHING FOR DETECTING FEATURES FROM HISTORICAL MAPS

Xue Xia [1*], Magnus Heitzler [1], Lorenz Hurni [1]

[1] ETH Zurich, Institute of Cartography and Geoinformation, 8093 Zurich, Switzerland - (xiaxue, hmagnus, lhurni)@ethz.ch

KEY WORDS: Template Matching, Convolutional Neural Networks, VGG19, Autoencoders, Feature Detection.

ABSTRACT:

Efficiently detecting features from historical maps is a challenging task due to its inconsistent manual scribbling styles and the lack of large scale labelled training data. To tackle this issue, this paper proposes an automatic feature detection pipeline utilizing CNN-based template matching (TM), which can lead to efficient feature extraction with minimal input, i.e. one single template. Three CNN-based TM models equipped with different feature extractors are investigated and compared in this research, namely pre-trained VGG19 CNNs, autoencoders, and the combination of both. Experiments conducted on six tiles of the Swiss Old National Map demonstrate that the combined architecture achieves the best result in wetlands detection, resulting in a mean intersection over union (*IoU*) of 69% and an average *F1* measure of 82%.

1. INTRODUCTION

Historical topographic maps represent the earth's surface in the past with rich geographic features and high geospatial accuracy. It can provide valuable information enabling large scale time series analysis for diverse scientific domains, including ecology, urban planning, transportation, natural hazards, archaeology etc. (Heitzler and Hurni, 2019). Precisely extracting geographic features from historical maps in a modern geodata format is of utmost importance for breaking down barriers to accessing these information. This task is typically achieved by manually digitizing the map content (Chiang et al., 2020). Due to the huge cost of time and money needed for manual digitization, automated feature extraction methods are being actively exploited by researchers in recent years. Specifically, deep convolutional neural networks (CNNs) based models prove to be highly efficient. Heitzler and Hurni (2020) trained an ensemble of U-Nets to detect building footprints from Swiss historical maps and achieved an average intersection over union of 88.2% and an average precision of 98.55%. Promising results have also been achieved by Uhl et al. (2020), who applied CNNs to extract human settlement patterns from United States Geological Survey historical topographic maps. The success of CNNs lies in their capability of extracting representative spectral and geometric features from raw input. However, CNNs based methods are often claimed to be data-hungry. Large amounts of labelled training data need to be generated in advance, which is impractical in many cases.

Template matching (TM) is another commonly used technique in computer vision applications. Classic TM finds the match between a template and the target image by comparing raw pixel values within a searching window using various similarity metrics, such as sum-of-squared-differences (SSD) and normalized cross-correlation (NCC). Unlike CNNs, no annotated data is needed during this procedure. However, classic TM does not handle well complex cases when the transformation between the template and the target image is non-rigid or contains occlusions, which is quite common in real-life (Talmi et al., 2017). In order to provide more tolerance in TM, recent methods perform TM using deep features produced by pre-trained CNNs, instead of raw colour pixels (Cheng et al., 2019; Gao and Spratling, 2020). These CNN-based TM methods can take advantage of the feature extraction ability of CNNs to make TM more discriminative and also more tolerant to appearance change, while not at the cost of tedious data annotation task. One successful example is called QATM (quality-aware template matching), developed by Cheng et al. in 2019. QATM uses deep features extracted from *conv1_2* and *conv3_4* layers of a pre-trained VGG19 CNN. A soft-ranking of the similarity measure between the feature representation of the template patch and the target image patch among all matching pairs is used to assess the matching quality. In this way, the uniqueness of each pair can be taken into consideration. Experimental analysis shows that QATM outperforms state-of-the-art TM methods, including Best-Buddies-Similarity (BBS) (Dekel et al., 2015), Deformable Diversity Similarity (DDIS) (Talmi et al., 2017) and Co-occurrence based template matching (CoTM) (Kat et al., 2018), plus the classic TM using SSD and NCC, on standard template matching benchmarks. Therefore, we choose QATM as our baseline model in this research.

The main idea of CNN-based TM is to change the feature space in which the comparison between the template and the image is performed from the high-dimensional raw observations to a lower-dimensional representation. Except for pre-trained CNNs, autoencoders can also learn useful representations of data without any label. Besides, features learned from the first few layers of CNNs pretrained on a large and versatile dataset (ImageNet) are generic features such as lines and strokes, while features learned from autoencoders are more tailored to input data (Guérin et al., 2021). This is due to the fact that autoencoders are trained to reconstruct the original input from the encoded low-dimensional representation. Therefore, the fusion of generic features extracted by pre-trained CNNs and specific features extracted by autoencoders might provide a better picture of the targeted feature space. In this study, we introduce autoencoders to the QATM framework by concatenating the feature maps generated by autoencoders and pre-trained CNNs. We use the modified framework to detect

* Corresponding author

wetlands from historical maps. The main contributions of this study are the following:

1) We introduce the combination of generic features extracted by pre-trained CNNs and specific features generated by autoencoders to construct better feature representations;
2) We design a pipeline to detect wetlands from historical maps automatically without the need of labelled training data.

2. METHODOLOGY

2.1 Study area

In this research, we aim to detect wetlands from the Swiss Old National Map[1] series covering the densely populated Swiss Plateau area from 1952 to 1994. Wetlands are of high ecological value, but they are also threatened by the expansion of cities, transport infrastructure and agricultural land. Revealing the evolution of wetlands is essential to find out the impact of human activities to wetland ecosystems, as well as to guide future sustainable development.

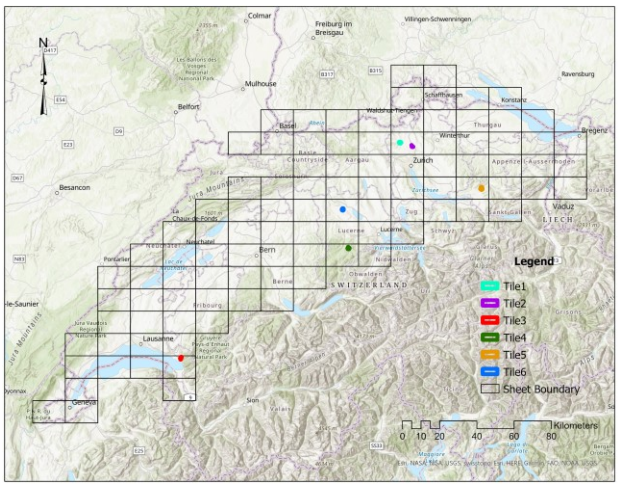

Figure 1. Extent of the study area and the location of the selected tiles for experimental analysis.

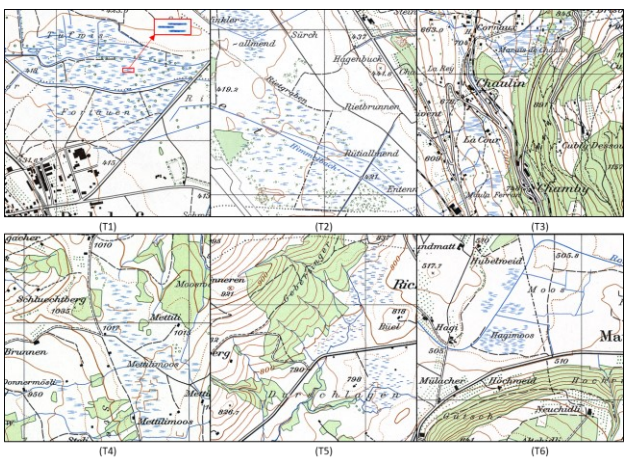

Figure 2. The selected map tiles and the template (in red bounding box from T1).

Figure 1 shows the extent of Swiss Old National Map sheets in Swiss Plateau area. Six tiles of 1000×1000 pixels were selected for our experimental analysis. These six map tiles spread across the whole study area and are from different years with slightly different scribbling styles. As shown in Figure 2, some wetland signatures have long and dense strokes, while some are short and sparse. We use these six different tiles to test the generalization ability of our model. The wetland signature template is cropped from tile 1 with a size of 20×48 pixels (Figure 2). It is aimed to detect wetlands from all these six different tiles with a single template. The underlying intention is to detect as much wetlands as possible with as little input as possible.

2.2 Overall Methodology

The overall methodology is presented in Figure 3, including three major steps, i.e. wetland signatures detection, wetland boundaries generation and accuracy assessment.

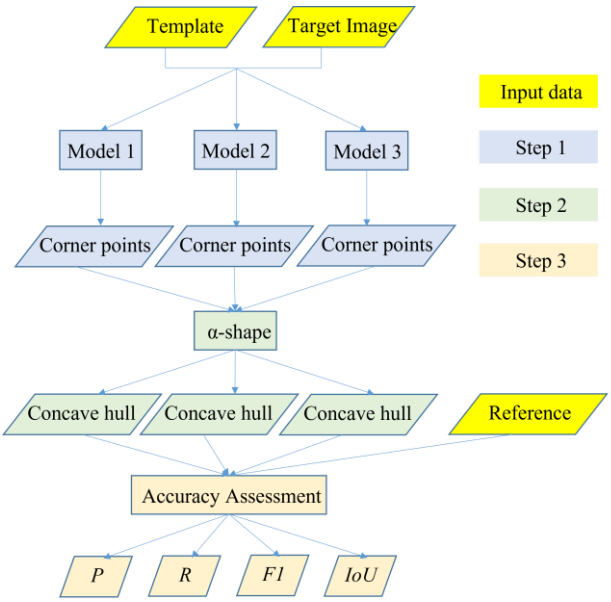

Figure 3. Overall methodology.

Step 1 is wetland signatures detection. In order to detect wetland signatures, we applied and compared three TM models. The first one is QATM developed by Cheng et al. (2019). This is one of the state-of-the-art CNN-based TM models using pre-trained VGG19 CNNs to construct the feature space for comparison. We use this model as our baseline model. In the second model, the feature space generated by pre-trained VGG19 is replaced with the ones generated by an autoencoder, while in the third model, a combination of feature maps from both networks (VGG19 and autoencoders) is adopted. The corner coordinates of the bounding boxes indicating matching locations are exported for further analysis. In step 2, we take these corner coordinates as input and create their concave hull with α-shape to approximate the wetland boundaries. The generated boundaries are finally compared with the reference data to evaluate the accuracy in step 3. In the following sections, we will give a detailed explanation for each step.

2.3 Step 1: Wetland Signatures Detection

We detect wetlands by taking one wetland signature as the template (Figure 2) and try to find all of the matching locations in the target map tiles. It is a challenging task as different map tiles have different drawing styles resulting from its manual inconsistent production process.

[1] https://www.swisstopo.admin.ch/en/geodata/maps/historical/old-national-maps.html

The TM models used in this research are modified from QATM (Cheng et al., 2019). Given a template image *T* and a search image *S*, their feature representations are first generated using a feature extraction model, i.e. a pre-trained VGG19 CNN. More precisely, the output feature map of *conv1_2* layer is resized and concatenated with the output feature map of *conv3_4* layer as the constructed feature representation. Table 1 presents the convolutional blocks of VGG19, excluding the top dense layers. Given an input image with a size of $H \times W \times C$, in which *H*, *W* and *C* namely represent the height, width and channels of the image, the output feature map of *Conv1_2* layer has a size of $H \times W \times 64$, and the output size of *Conv3_4* layer is $H/4 \times W/4 \times 256$. Therefore, the final feature representation has a size of $H/4 \times W/4 \times 320$.

After that, a predefined similarity measure between the feature representations of the template patch and the search image patch is calculated. In their paper, cosine similarity is used to assess the patch-wise similarity. However, the final matching quality score is not defined by this measure. Alternatively, a likelihood function is introduced to define the soft-ranking of the current pairs compared to all other pairs in terms of matching quality.

Layer	Type	Output Shape
Input	Image input	*(None, 1000, 1000, 3)*
Conv1_1	Convolution	*(None, 1000, 1000, 64)*
Conv1_2	Convolution	*(None, 1000, 1000, 64)*
Pool1	Max Pooling	*(None, 500, 500, 64)*
Conv2_1	Convolution	*(None, 500, 500,128)*
Conv2_2	Convolution	*(None, 500, 500, 128)*
Pool2	Max Pooling	*(None, 250, 250, 128)*
Conv3_1	Convolution	*(None, 250, 250, 256)*
Conv3_2	Convolution	*(None, 250, 250, 256)*
Conv3_3	Convolution	*(None, 250, 250, 256)*
Conv3_4	Convolution	*(None, 250, 250, 256)*
Pool3	Max Pooling	*(None, 125, 125, 256)*
Conv4_1	Convolution	*(None, 125, 125, 512)*
Conv4_2	Convolution	*(None, 125, 125, 512)*
Conv4_3	Convolution	*(None, 125, 125, 512)*
Conv4_4	Convolution	*(None, 125, 125, 512)*
Pool4	Max Pooling	*(None, 62, 62, 512)*
Conv5_1	Convolution	*(None, 62, 62, 512)*
Conv5_2	Convolution	*(None, 62, 62, 512)*
Conv5_3	Convolution	*(None, 62, 62, 512)*
Conv5_4	Convolution	*(None, 62, 62, 512)*
Pool5	Max Pooling	*(None, 31, 31, 512)*

Table 1. The convolutional blocks of VGG19, taking an input image with a size of $1000 \times 1000 \times 3$ as an example to calculate the output shape.

In this study, we propose two variants of QATM by changing its feature extraction module. In the first one, this module is totally replaced by an autoencoder trained on the template patch, and in the second one, a combination of the original pre-trained VGG19 and the autoencoder is applied via concatenating the feature maps produced by both networks. The architecture of the autoencoder used in both variants is presented in Figure 4. The encoder part consists of two convolutional layers and two max-pooling layers, which compress the input into a latent space representation. The decoder part consists of two transposed convolution layers and one convolutional layer to reconstruct the compressed data. As we have only one template patch, the autoencoder is trained on this single input image for 500 epochs. The reconstructed image is also visualized in Figure 4, which highly resembles the template. After training, we take the encoder part to generate feature representations for both the template and the search image. Given an input image with a size of $H \times W \times C$, the feature map generated by the encoder has a size of $H/4 \times W/4 \times 64$. As a result, the final feature representation of the first variant using autoencoders alone as its feature extractor has a size of $H/4 \times W/4 \times 64$. The second variant concatenates the feature maps produced by VGG19, thus receiving a feature representation of size $H/4 \times W/4 \times 384$.

The matching quality score is calculated following the original settings in QATM. By thresholding the matching quality score, we get the final matching locations of the wetland signatures and forward them to the next step.

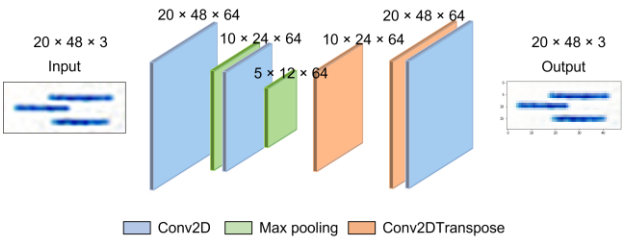

Figure 4. Architecture of the autoencoder. The output shape of each layer is also presented.

2.4 Step 2: Wetland Boundaries Generation

The output from step 1 are the corner coordinates of the bounding boxes showing the matching locations of the wetland signature. We approximate the wetlands' bounding hulls on these corner points using α-shapes. An α-shape is a subgraph of the closest point or furthest point Delaunay triangulation (Edelsbrunner et al., 1983). It can identify the area occupied by a set of points using a "fine shape" or a "crude shape" when different α values are applied (Asaeedi et al., 2017). We fine-tuned the α values via visual interpretation to generate the wetland boundaries.

2.5 Step 3: Accuracy Assessment

Four accuracy measures are assessed to compare the bounding hulls against the ground truth, namely Precision (*P*), Recall (*R*), F1-score (*F1*) and Intersection over Union (*IoU*). Precision measures the ratio of correctly detected wetlands to all detected wetlands, while recall indicates the ratio of correctly detected wetlands to all wetlands in reference. *F1* is the harmonic mean between precision and recall, which can be regarded as an overall accuracy measure. *IoU* is the percentage of the intersection area over the union area of the predicted wetlands and the ground truth.

3. RESULTS

We compared experimental results of the three models on six testing tiles respectively. As the main difference of the three models is the feature extraction module, we distinguish them by their feature extraction module type, namely VGG19 (in original QATM, model 1), autoencoders (AEC, model 2), VGG19 plus autoencoders (VGG19 + AEC, model 3).

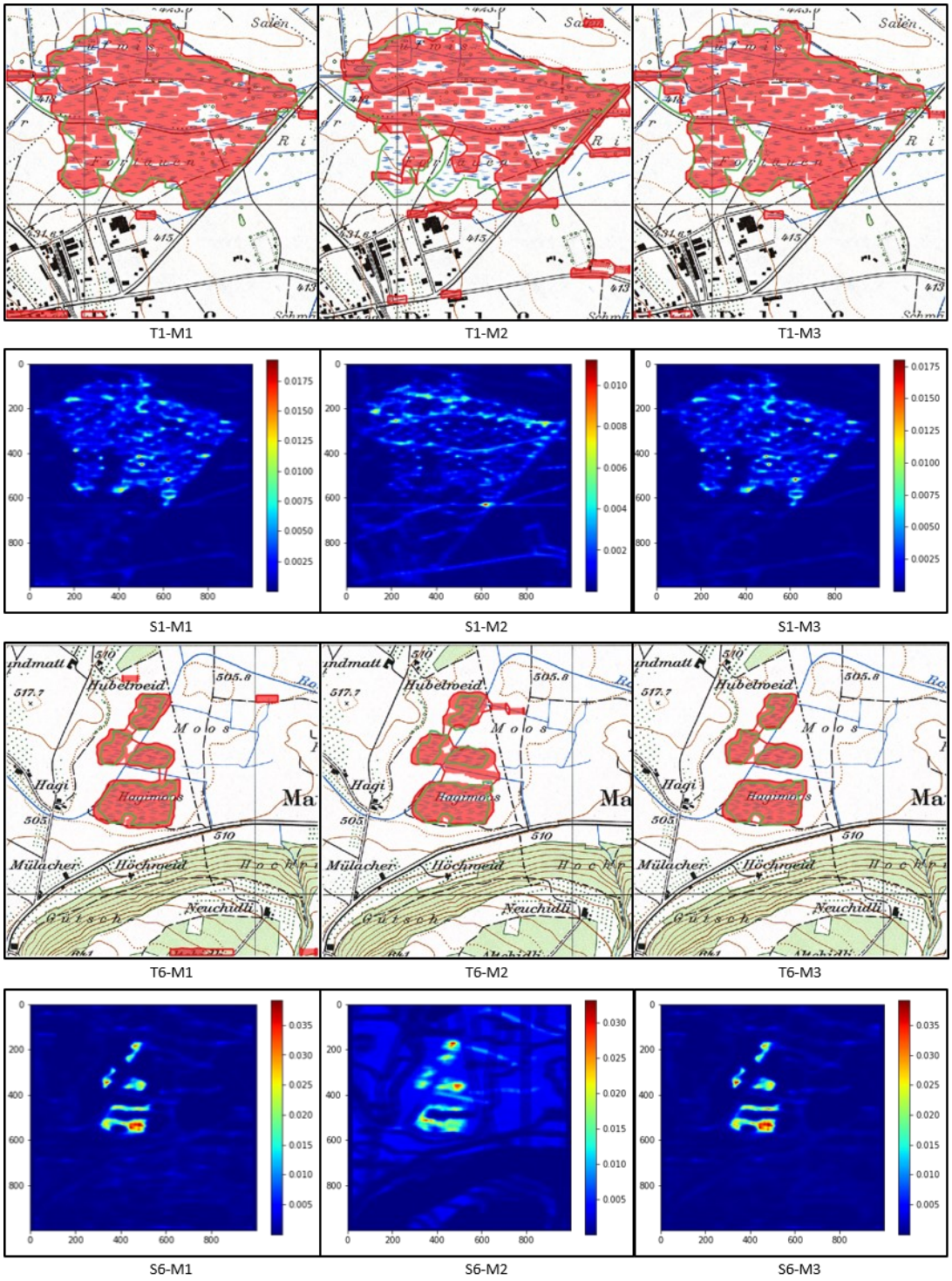

Figure 5. The detection results of Tile 1 and Tile 6 with their similarity score maps. T1-M1 refers to the result of Tile 1 generated by Model1, and S1-M1 is its corresponding similarity score map. The rest of the coded captions can be understood accordingly. Detected wetland signatures are marked in red bounding boxes; Red boundary is the generated wetland boundary and the green boundary marks the ground truth.

Figure 5 and Figure 6 compare the bounding boxes of the detected wetland signatures and the final generated wetland polygons produced by the three models with the corresponding reference. Based on visual interpretation, almost for all testing tiles, the shape of the bounding hull can well identify the area covered by the bounding boxes when an α value of 0.04 is applied. We also picked two random tiles and visualized their similarity score maps in Figure 5. Generally speaking, despite the differences in drawing styles, most wetland signatures from the six testing tiles are successfully detected by the three models, indicating the good generalization ability of the models. From all six tiles, the performance of model 3 ranks first due to its high detection rate and low error rate, followed by model 1, which detects similar amounts of wetland signatures as model 3, but with more obvious false positives. The performance of model 2 is in general the worst. This result shows that the

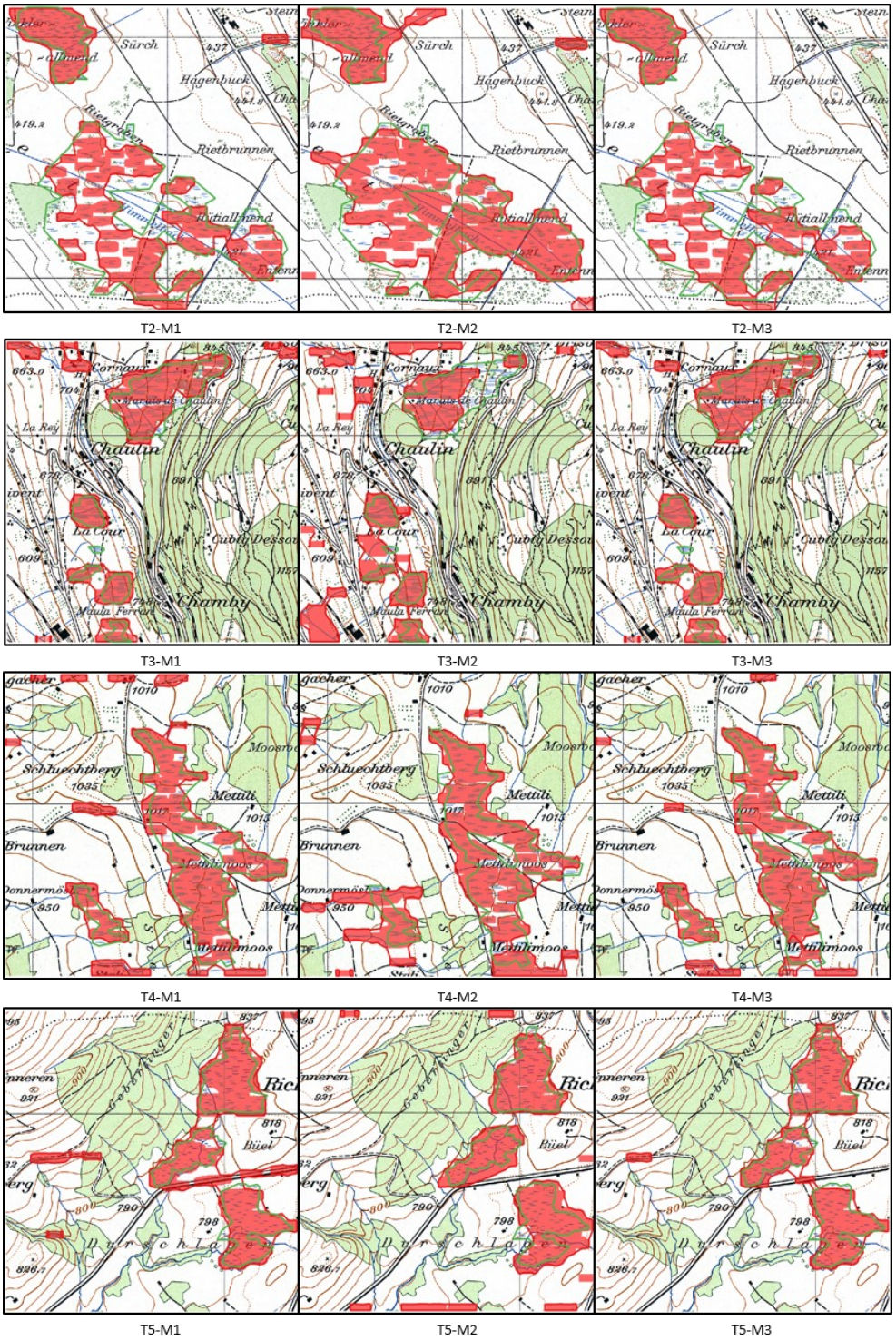

Figure 6. The detection results of Tile 2, Tile 3, Tile 4, and Tile 5. T2-M1 refers to the result of Tile 2 generated by Model1, and the rest of the coded captions can be understood accordingly. Detected wetland signatures are marked in red bounding boxes; Red boundary is the generated wetland boundary and the green boundary marks the ground truth.

combination of generic features and specific features can better distinguish the target from the rest. Evidence can also be witnessed from the similarity maps (Figure 5). Similarity maps produced by model 3 have salient peaks and a steady background, indicating regions with and without wetland signatures respectively, while those produced by model 1 and model 2 have messier backgrounds and thus making it harder to identify the target. Although the visual results of model 2 is the worst, we still cannot deny the performance of autoencoders as it is trained on only one image, which is the template in our case. By contrast, pre-trained VGG19 is trained on the ImageNet database that contains a million images (Bansal et al., 2021). In summary, integrating autoencoders in the original QATM framework which only employs pre-trained VGG19 as its feature extractor improves the detection result, resulting in better aligned wetland polygons with less false positives.

Table 2 shows the quantitative results obtained by the three models. The mean *IoU* of the six testing tiles achieved by model 3 is 69%, against 66% of model 1 and 59% of model 2, demonstrating that the wetland polygons detected by model 3 align better with ground truth than the other two models. Model 3 also outperforms model 1 and 2 in terms of *F1* measure, with an average value of 82% against 79% and 74% respectively, indicating its better overall performance. More concretely, compared to model 1, model 3 has higher average precision while slightly lower average recall. This shows that model 3 improves the overall performance mainly through preventing false positives. Moreover, all three models achieve the best result on tile 1 amongst all testing tiles. This is because the template is extracted from tile 1.

Accuracy		T1	T2	T3	T4	T5	T6	Mean
Model1	*P*	0.87	0.88	0.64	0.65	0.62	0.60	0.71
	R	0.97	0.72	0.92	0.94	0.97	0.99	0.92
	F1	0.92	0.79	0.76	0.77	0.75	0.75	0.79
	IoU	0.85	0.66	0.61	0.62	0.61	0.59	0.66
Model2	*P*	0.80	0.74	0.46	0.60	0.62	0.60	0.64
	R	0.85	0.88	0.71	0.92	0.96	0.96	0.88
	F1	0.83	0.80	0.56	0.73	0.76	0.74	0.74
	IoU	0.71	0.67	0.39	0.57	0.61	0.58	0.59
Model3	*P*	0.89	0.89	0.67	0.69	0.71	0.71	0.76
	R	0.96	0.71	0.90	0.90	0.96	0.98	0.90
	F1	0.92	0.79	0.77	0.78	0.82	0.82	0.82
	IoU	0.86	0.65	0.62	0.64	0.69	0.70	0.69

Table 2. Quantitative results of the testing tiles.

4. DISCUSSION

The findings of this study clearly show that using combined feature maps of pre-trained VGG19 and autoencoders can build a feature space better representing the template and the target image, resulting in better performance in wetlands detection. One explanation for the performance gain is the superiority of integrating generic features learnt from large datasets with specific features learnt from application-specific data. There are some limitations of this study, which are also the focuses of our future research:

1) Tuning the weight of generic features and specific features in the constructed feature space. In our experimental settings, the feature map generated by pre-trained VGG19 has 320 channels, while that generated by autoencoders has only 64 channels. In other words, generic features account for a larger proportion than specific features in the integrated feature space. We tried to increase the feature map channels by directly increasing the numbers of the filters in the corresponding convolutional layers of the autoencoder. However, we only ended up with many empty layers (pixel values are all zero) in the output feature map. Considering we have only one input image to train the autoencoder, simply increasing the number of filters cannot add up useful features to the output. In the future, we can try to use more complex architectures for the autoencoder or to train the autoencoder with more input images (more templates).

2) Try to use the model in an iterative manner. This study is limited by using only one template to train the autoencoder, resulting in highly overfitting models. However, this study can also be regarded as the starting step of an iterative experimental procedure. We start from one template. Use it to train an autoencoder and run the detection model to return the coordinates of the detected patches. We extract these patches and use them as our new templates. After that, we iterate the first step. Use the new templates to train a better autoencoder, and to run the detection model again to find matching locations of the multiple templates. By using the model iteratively, we might approximate the goal of detecting all targets.

5. CONCLUSIONS

This paper proposes a novel pipeline for wetlands detection from historical maps using CNN-based TM and α-shape. We get rid of the extensive data annotation task as is often required by many CNN-based methods via employing pre-trained networks and self-supervised autoencoders. Moreover, we compared three different feature extraction models in the CNN-based TM framework. Experimental analysis shows that combining pre-trained CNNs and autoencoders as the feature extractor achieves the best result. Aspects that need to be further investigated are the weight distribution when combining feature maps from pre-trained CNNs and autoencoders and the iterative utilization of the proposed method to improve accuracy.

REFERENCES

Asaeedi, S., Didehvar, F., Mohades, A., 2017. α-Concave hull, a generalization of convex hull. Theor. Comput. Sci. 702, 48–59. https://doi.org/10.1016/j.tcs.2017.08.014

Bansal, M., Kumar, M., Sachdeva, M., Mittal, A., 2021. Transfer learning for image classification using VGG19: Caltech-101 image data set. J. Ambient Intell. Humaniz. Comput. https://doi.org/10.1007/s12652-021-03488-z

Cheng, J., Wu, Y., Abdalmageed, W., Natarajan, P., 2019. QATM: Quality-aware template matching for deep learning. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 2019-June, 11545–11554. https://doi.org/10.1109/CVPR.2019.01182

Chiang, Y.-Y., Duan, W., Leyk, S., Uhl, J.H., Knoblock, C.A., 2020. Using Historical Maps in Scientific Studies, SpringerBriefs in Geography. Springer International Publishing, Cham. https://doi.org/10.1007/978-3-319-66908-3

Dekel, T., Oron, S., Rubinstein, M., Avidan, S., Freeman, W.T., 2015. Best-Buddies Similarity for robust template matching. Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. 07-12-June, 2021–2029. https://doi.org/10.1109/CVPR.2015.7298813

Edelsbrunner, H., Kirkpatrick, D., Seidel, R., 1983. On the shape of a set of points in the plane. IEEE Trans. Inf. Theory 29, 551–559. https://doi.org/10.1109/TIT.1983.1056714

Gao, B., Spratling, M.W., 2020. Robust Template Matching via Hierarchical Convolutional Features from a Shape Biased CNN. ArXiv abs/2007.1, 1–11.

Guérin, J., Thiery, S., Nyiri, E., Gibaru, O., Boots, B., 2021. Combining pretrained CNN feature extractors to enhance clustering of complex natural images. Neurocomputing 423, 551–571. https://doi.org/10.1016/j.neucom.2020.10.068

Heitzler, M., Hurni, L., 2020. Cartographic reconstruction of building footprints from historical maps: A study on the Swiss Siegfried map. Trans. GIS 24, 442–461. https://doi.org/10.1111/tgis.12610

Heitzler, M., Hurni, L., 2019. Unlocking the Geospatial Past with Deep Learning-Establishing a Hub for Historical Map Data in Switzerland 15–20. https://doi.org/10.5194/ica-abs-1-110-2019

Kat, R., Jevnisek, R., Avidan, S., 2018. Matching Pixels Using Co-occurrence Statistics, in: 2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition. IEEE, pp. 1751–1759. https://doi.org/10.1109/CVPR.2018.00188

Talmi, I., Mechrez, R., Zelnik-Manor, L., 2017. Template

Matching with Deformable Diversity Similarity, in: 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR). pp. 1311–1319.

Uhl, J.H., Leyk, S., Chiang, Y.-Y., Duan, W., Knoblock, C.A., 2020. Automated Extraction of Human Settlement Patterns From Historical Topographic Map Series Using Weakly Supervised Convolutional Neural Networks. IEEE Access 8, 6978–6996. https://doi.org/10.1109/ACCESS.2019.2963213

Zhang, X., Yu, F.X., Karaman, S., Zhang, W., Chang, S.-F., 2018. Heated-Up Softmax Embedding. ArXiv abs/1809.0, 1–11.