



## Polyline simplification using a region proposal network integrating raster and vector features

Baode Jiang, Shaofen Xu & Zhiwei Li

To cite this article: Baode Jiang, Shaofen Xu & Zhiwei Li (2023) Polyline simplification using a region proposal network integrating raster and vector features, GIScience & Remote Sensing, 60:1, 2275427, DOI: [10.1080/15481603.2023.2275427](https://doi.org/10.1080/15481603.2023.2275427)

To link to this article: <https://doi.org/10.1080/15481603.2023.2275427>



© 2023 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.



Published online: 30 Oct 2023.



Submit your article to this journal [↗](#)



View related articles [↗](#)



View Crossmark data [↗](#)

# Polyline simplification using a region proposal network integrating raster and vector features

Baode Jiang<sup>a,b</sup>, Shaofen Xu<sup>b,c</sup> and Zhiwei Li<sup>c</sup>

<sup>a</sup>School of Computer Science, China University of Geosciences, Wuhan, China; <sup>b</sup>National Engineering Research Center of Geographic Information System, China University of Geosciences, Wuhan, China; <sup>c</sup>Department of Land Surveying and Geo-Informatics, The Hong Kong Polytechnic University, Hong Kong, China

## ABSTRACT

Polyline simplification is crucial for cartography and spatial database management. In recent decades, various rule-based algorithms for vector polyline simplification have been proposed. However, most existing algorithms lack parameter self-adaptive capabilities and require repeated manual parameter adjustments when applied to different polylines. While deep-learning-based methods have recently been introduced for raster polyline image simplification, they cannot achieve end-to-end simplification when the input data and output results are vector polylines. This paper proposes a new deep-learning-based method for vector polyline simplification by integrating both the vector and raster features of the polyline. Specifically, a deep separable convolutional residual neural network was first used to extract the convolutional features of each image. Then, the region proposal network was modified to generate proposal boxes using vector coordinates, and these proposal boxes were used to locate the convolutional feature maps of bends. Finally, convolutional feature maps were fed into a binary classification network to identify unimportant vertices that should be omitted for vector polyline simplification. The experimental results indicated that the proposed method can exploit raster and vector features to achieve automatic and effective polyline simplification without prior map generalization knowledge and manual settings of rules and parameters. The polyline simplification results of the proposed method have a higher compression ratio of coordinate points and lower shape deformation and deviation than the results generated by the classic Wang and Müller (WM) algorithm and Support Vector Machine (SVM) based algorithm, which shows the potential of the proposed method for future applications in map generalization.

## ARTICLE HISTORY

Received 16 December 2022  
Accepted 20 October 2023

## KEYWORDS

Map generalization; polyline simplification; deep learning; region proposal network

## 1. Introduction

Map generalization plays an essential role in generating multi-scale maps, managing spatial data, and transmitting map data online (Ai et al. 2017; Liu et al. 2016; Vrotsou et al. 2015). With rapid developments in the automatic mapping of remote sensing images, the problem of simplifying the vector polylines obtained from edge detection and edge tracking has led to new requirements for generalization technology (Chen et al. 2020). Most classical polyline simplification algorithms are based on computational geometry techniques and cartographic rules (Douglas and Peucker 1973; Li and Openshaw 1992; Stanislawski et al. 2014; Visvalingam and Whyatt 1993; Wang and Müller 1998). However, these rule-based methods are often facing limitation in performance, particularly when the man-made rules have to be adapted to different polylines (Du et al. 2021; Kent 2017).

Recently, some machine-learning algorithms have provided a new generalization paradigm that can learn abstract map generalization rules from existing simplification cases without manual parameter configurations (Jiang 2003; Karsznia and Weibel 2018; Lee et al. 2017). For example, back propagation neural networks (Cheng et al. 2013), decision trees, genetic algorithms (Karsznia and Sielicka 2020), and support vector machines (SVM) (Duan et al. 2020) are the most commonly used machine-learning methods for map generalization. Although these machine-learning-based methods, which can be generally classified into vertex- and image-based categories, can learn some simplification rules, both categories have their own shortcomings. Vertex-based methods cannot utilize image features, whereas image-based methods have difficulty preserving original coordinates (Du et al. 2022). Further,

**CONTACT** Shaofen Xu  [xvshaofen@cug.edu.cn](mailto:xvshaofen@cug.edu.cn)

© 2023 The Author(s). Published by Informa UK Limited, trading as Taylor & Francis Group.

This is an Open Access article distributed under the terms of the Creative Commons Attribution-NonCommercial License (<http://creativecommons.org/licenses/by-nc/4.0/>), which permits unrestricted non-commercial use, distribution, and reproduction in any medium, provided the original work is properly cited. The terms on which this article has been published allow the posting of the Accepted Manuscript in a repository by the author(s) or with their consent.

image-based methods are based on feature engineering, which cannot achieve end-to-end data-driven training.

As an end-to-end solution, deep-learning-based methods can operate adaptively according to original data distributions without prior human knowledge. Such models have been successfully applied in the field of computer vision for image classification, semantic segmentation, and object detection (LeCun, Bengio, and Hinton 2015; Yu and Chen 2022). Recently, deep-learning-based methods have been applied to geospatial domains, such as in the automatic extraction of feature targets on maps and remote sensing images using deep convolutional models (Li and Hsu 2020), map style migration using generative adversarial networks (GANs) (Kang, Gao, and Roth 2019), and code protection of geographic location information using synthetic information (Rao et al. 2020). Considering that deep-learning-based methods demonstrate great promise in improving the efficiency and automaticity of map generalization (Ai 2021), some studies have applied deep-learning-based methods to automatic map generalization, such as building simplification and aggregation and road network simplification (Yan et al. 2020; Zhou and Li 2017). However, most studies can only deal with raster data and end up neglecting vector map data. Additionally, these existing deep-learning-based line simplification methods can only obtain raster simplification results, which cannot be directly used in vector map cartography and stored in vector spatial databases.

To address these issues, a vector polyline simplification method based on the region proposal network (RPN) is proposed herein. It integrates raster and vector features to obtain an end-to-end vector polylines simplification. Here, first, the vector polyline map was converted to a raster map and fed into a deep separable convolutional residual neural network to extract the convolutional features from the map. Second, the proposal generator of the RPN was modified to generate proposal boxes using the minimum bounding rectangles of vector coordinates, which were then used to locate the bend feature maps. Finally, these feature maps with different sizes were unified by the regions of interest (ROI)-pooling layer and fed into a binary classification network to obtain a vector polyline simplification result.

The main contributions of our approach include the following three aspects. First, an end-to-end trainable neural network based on the RPN is proposed for implementing polyline simplification. Compared with traditional rule-based methods, the method does not require prior map generalization knowledge or manual settings of rules and parameters. Second, in contrast to the existing RPN that generates proposal boxes from each pixel of the image, a new region proposal box generation method for vector polylines based on the coordinates of polyline vertices, which generates fewer proposal boxes and can integrate vector and raster features of the polyline, is proposed. Finally, compared with most of the existing deep-learning-based polyline simplification methods that can only deal with raster data and obtain the simplification result in a raster format, the proposed method can implement vector polyline simplification and obtain a vector format result, which avoids further raster-to-vector conversion and saves time consumption in polyline simplification.

## 2. Related studies

Since the 1970s, several classical line simplification algorithms have been proposed. For example, Douglas and Peucker (1973) (DP) developed the famous DP algorithm, which determines the trade-off between points based on the perpendicular distance between a vertex and line connecting the start and end points. Li and Openshaw (1992) introduced the natural rule concept of the “smallest visible object” and achieved local adaptive linear simplification based on the Li – Openshaw algorithm. Visvalingam and Whyatt (1993) considered the “effective area” of points to achieve the progressive simplification of polylines. Wang and Müller (1998) (WM) proposed that the overall structure of the line could be reasonably preserved by decomposing the line structure into a series of line bends and simplifying the line using rules such as elimination, combination, and amplification. Based on the above method, some researchers have combined or improved individual simplification operators to improve the performance of algorithms (Park and Yu 2010; Zhou and Li 2017). Additionally, the Voronoi diagram and Delaunay triangulations have been proposed as methods to construct polylines and achieve line simplification using predefined thresholds (Ai et al. 2017).

Most of the above classical line simplification algorithms depend on certain mapping rules developed by cartographic experts, which are hard to adapt to different contexts. The rapid development of deep learning technology, which has numerous successful applications in digital image processing (LeCun, Bengio, and Hinton 2015), map style transfer (Kang, Gao, and Roth 2019), remote sensing image interpretation (Li and Hsu 2020), provides new development opportunities for map generalization. Based on deep learning methods, the automatic capturing of image features such as size, granularity, and shape, as well as the exploration of deep information hidden in images, can effectively simulate the implementation of map generalization tasks. In addition, the development of cartography over the years has provided a large-scale comprehensive case sample library for deep-learning-based methods that can be used for training and learning (Courtial, Touya, and Zhang 2023). This has made it possible to combine deep learning with cartography, which was previously not feasible.

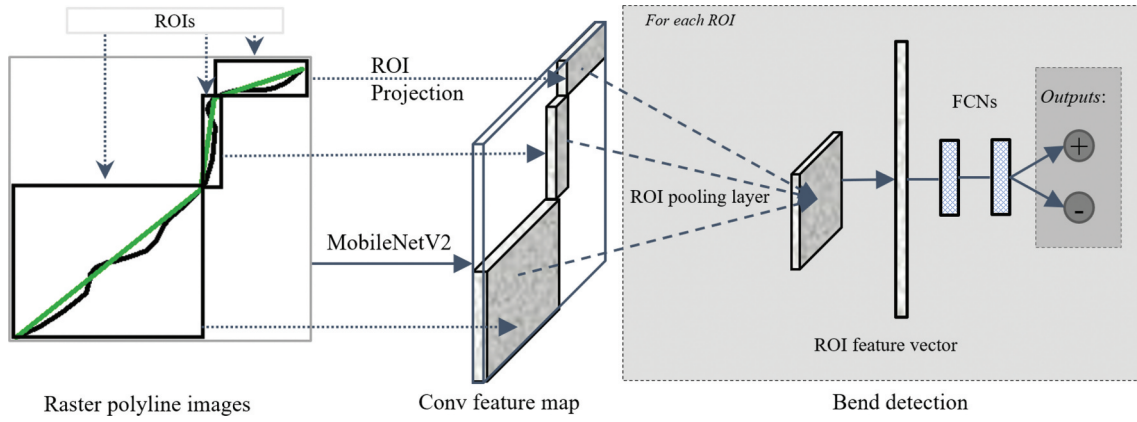
Currently, automatic map generalization based on deep learning remains in its infancy and is mainly applied to building and settlement generalization (Touya, Zhang, and Lokhat 2019; Yang et al. 2022). For example, Sester et al. (2018) introduced a deep-learning-based method in building generalization for the first time, using binary images as training data to train the U-net (Ronneberger, Fischer, and Brox 2015) model and obtained simplified building images. Feng et al. (2019) extended this study by comparing the building simplification performance using U-net, residual U-net, and a GAN at different target map scales and discovered that residual U-net demonstrated the best generalization performance. Courtial et al. (2021) used a GAN to achieve map generalization of buildings, roads, and rivers in densely populated urban areas. However, the generated map tiles suffered from problems such as overlapping elements and loss of information.

Certain polyline simplification studies based on deep learning have also been conducted. For example, Courtial et al. (2020) explored the possibility of applying U-net to mountain road generalization. However, the simplified results were in a raster format with rough edges and holes. Later, they further studied the application of GANs for mountain road simplification (Courtial, Touya, and Zhang 2023).

Although they proposed some graphic constraints to improve the generated image quality (Courtial, Touya, and Zhang 2022), important local issues persisted, and the resulting images could not be used as topographic maps. Du et al. (2021) used both vector and raster features to omit vertices – they proposed a polyline tracking segmentation algorithm to generate training images for the Pix2Pix (Isola et al. 2017) model and obtain simplified images. However, they still needed to use the DP algorithm to simplify the vectorized polylines from the image results. The aforementioned studies were only able to simplify raster line features, and the generated results cannot be directly used. Currently, the vector data format remains the primary data format in map analysis and management. To address the issue of polyline simplification in the vector format, Yu and Chen (2022) developed an encoder – decoder network to generate multilevel simplified polylines from multiple hidden layers. However, the result of this method was too smooth to retain the original important feature coordinates, and the input and output data (a stacked autoencoder encoding) were set with a fixed length, which required the model structure to be changed when dealing with different line simplification scenarios. Therefore, an RPN-based method for vector polyline simplification that integrates both vector and raster features of the polyline is proposed in this paper. The proposed method is an end-to-end solution without man-made parameters, and the result of polyline simplification is directly in the vector format.

### 3. Method

The proposed method is an end-to-end model for polyline simplification, which was inspired by the concept of line simplification based on bend analysis (Wang and Müller 1998), i.e. if all the bends on a polyline could be detected and eliminated, automatic polyline simplification would be completed. The overall framework of the method, which comprises bend feature extraction, region proposal generation, and bend detection, is illustrated in Figure 1. First, a CNN was used to extract the convolutional feature map of the raster polyline. Second, the minimum bounding rectangles of bends on vector polylines were used to generate region proposals of the RPN as ROIs. Third, the bend features located by ROIs were unified by the ROI-pooling layer and fed into



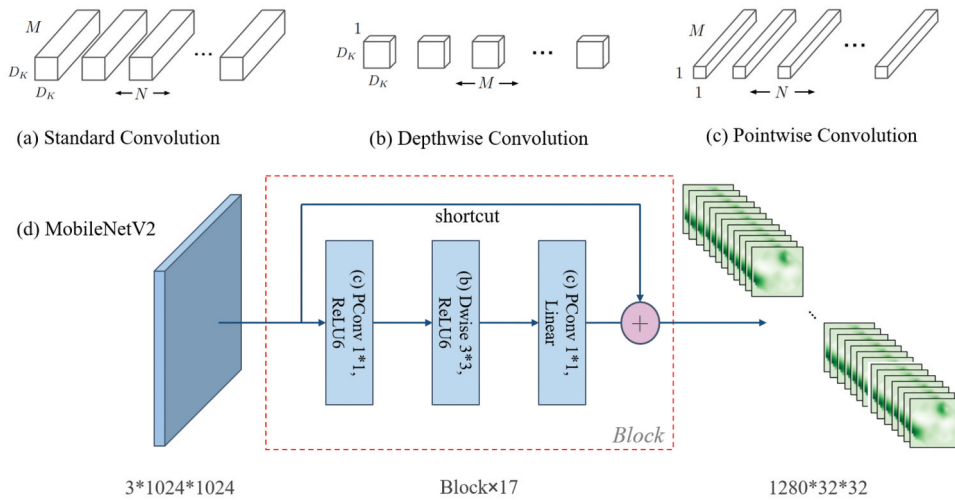
**Figure 1.** Illustration of the polyline simplification method based on the RPN. A raster polyline image and multiple region proposals are first input into a deep convolutional network to obtain the bend features. For each ROI, the extracted feature is pooled to the same size and then mapped by fully connected (FC) layers, and a *softmax* layer is used to determine the true proposal.

a binary classification network to identify unimportant bends that should be omitted. After eliminating the unimportant bends from the vector polylines, simplification results of polylines in a vector format were obtained. The details of the proposed method are specified as follows.

### 3.1. Bend feature extraction

To extract the bend shape features of a polyline using a CNN, the vector polylines were first converted to raster images and segmented into units with the same size as required. The images were then fed into a CNN for feature extraction. MobileNetV2 (Sandler et al. 2018) was chosen as the backbone CNN to extract

the features of the raster polylines. It has few parameters and is easy to train, which helps prevent overfitting when training the model. MobileNetV2 was proposed based on the depth-wise separable convolution. Compared with the standard convolution layer, it divides a standard convolution into a  $1 \times 1$  convolution, called point-wise convolution, and a depth-wise convolution. As shown in Figure 2, (a) is the standard convolution with a convolution size of  $D_k \times D_k \times M$  and an input data size of  $D_k \times N \times M$ . (b) is a depth-wise convolution that applies a single filter to  $M$  input channels, and (c) is an  $N$  point-wise convolution that applies a  $1 \times 1 \times M$  convolution to combine the channel outputs. Depth-wise separable convolution replaces (a) with (b) and (c) and adds a batch normalization layer



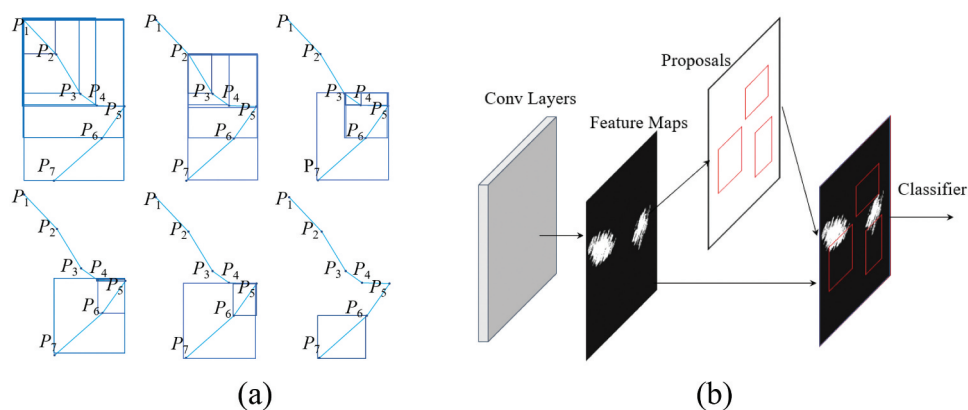
**Figure 2.** Illustration of depth-wise separable convolution. The depth-wise separable convolution is built by replacing (a) the standard convolution with (b) a depth-wise convolution and (c) a point-wise convolution. (d) network structure of MobileNetV2.

and a ReLU activation function layer after both (b) and (c), which can effectively reduce the size of the model and computational effort. (d) illustrates the structure of MobileNetV2, which adopts the structure of dilation, followed by convolution and compression. By inputting polyline images with a size of  $3 \times 1024 \times 1024$  into the model, convolutional feature maps of raster polylines with a size of  $1280 \times 32 \times 32$  can be obtained.

### 3.2. Region proposal generation

Inspired by the concept of target detection in computer vision, this paper introduces an RPN to locate bends on polylines. The RPN was first proposed in the Faster R-CNN (Ren et al. 2017); it traverses the feature map through a sliding window and generates  $k$  region proposals for each pixel (normally  $k$  is set to nine and comprises three different sizes and scales). Then, all the proposals are input into two layers: a regression layer that outputs the coordinates of the boxes and a classification layer that outputs the probability of being an object. However, when generating region proposals for line simplification in this manner, the following problems may arise: 1) The generated proposals with fixed sizes and scales cannot include all the bends on the polylines; 2) owing to the sparse information of the polyline feature map, generating proposals for each pixel results in a waste of computational resources; and 3) the generated proposals have no relationship with the vertices of the vector polyline and cannot integrate the vector features into the model when detecting bends. Therefore, a new region proposal generation method is proposed in this paper.

As the bend is formed by the vertex sequences on the vector polyline, the improved region proposal generation method generates proposals using the minimum bounding rectangles of the possible bend vertices of the original vector polyline. For example, assuming the points consist of the polyline are  $(P_1, P_2, P_3, \dots, P_n)$ , then the sequences of points that may form the bends are  $(P_1, P_2), (P_1, P_2, P_3), \dots, (P_1, P_2, \dots, P_n), (P_2, P_3), (P_2, P_3, P_4), \dots, (P_2, P_3, \dots, P_n), \dots, (P_{n-1}, P_n)$ , and the minimum bounding rectangles of these point sequences are considered as the proposals. An example of this new proposal generation method for a polyline with seven vertices is shown in Figure 3(a). This method has several advantages: 1) Generating proposals based on the vector polyline can build the corresponding relationship between the bending position of the raster image and the vector polyline. This is because the bounding rectangles of bends can be mapped to the convolutional feature maps, and the location of bends detected from the feature maps can be mapped to vector polylines. Thus, polyline simplification can be completed by deleting the unimportant bends according to the classified feature maps. 2) Considering that the number of polyline vertices contained in the raster image is much less than the number of pixels in the feature map, the proposed method can reduce the number of proposals and save computational costs. 3) The size and scale of the proposal depend on the minimum bounding rectangle of the possible bend's vertex sequences, which is more diverse and appropriate for the bend on the polyline and makes the extraction of bends on the polyline more accurate.



**Figure 3.** Illustration of region proposal generation and bend feature extraction. (a) region proposal generation for a polyline with seven vertices; (b) process of obtaining possible bend features based on generated proposals.

After generating the proposals using the bounding rectangles of point sequences of the polyline, the proposals are used to clip the possible bend features from the feature map and send them to the classifier to complete bend detection. The process for obtaining a possible bend is illustrated in Figure 3(b).

### 3.3. Bend detection

To use the classifier network to complete the detection of bends, the extracted features must first be unified to the same size. However, changing the size of the target feature maps through cropping or scaling inevitably changes its shape features, and thus, the ROI-pooling layer is adopted to unify the sizes of the feature maps. The ROI-pooling layer is a maximum pooling layer proposed by the Fast R-CNN (Girshick 2015) to solve the problem of inconsistent feature map sizes of ROIs or proposals. Figure 4(a) illustrates the ROI-pooling process, which is as follows: First, the feature map from the last convolution layer and the region proposals generated by the minimum bounding rectangles of the vertex sequences are obtained. Second, the proposals are mapped to the corresponding positions of the feature map according to the ratio of the original image to the feature map (after feature extraction in the previous convolutional layers, the image size was reduced 32 times, so the input proposals should also be reduced by 32 times). Third, the mapped area is divided into several sections of the same size. The number of sections is determined by the dimensional size of the output. In this study, the output size of the network structure was  $7 \times 7$ . Finally, the max pooling operation is performed in each section. After the bend feature maps of the same size are output, they are flattened and fed into two FC layers,

followed by a sigmoid layer to make a binary judgment with a cross-entropy loss function (equation 1). For feature maps judged to be positive, their corresponding proposals can be obtained, as shown in Figure 4(b), and the point sequences of bends that generate proposals can also be located on the vector polyline. Therefore, polyline simplification can be achieved by eliminating the sequence points (except for the first and last points of each sequence).

$$L_{cls} = -\log[p_i^* p_i + (1 - p_i^*)(1 - p_i)], \quad (1)$$

Where  $p_i$  is the true binary label,  $p_i^*$  is the probability predicted by the anchor as a target.

## 4. Experiment results and analysis

### 4.1. Experiment data

The experiments in this study were conducted on the coastline dataset GSHHG (Wessel and Smith 1996) (<http://www.soest.hawaii.edu/wessel/gshhg>). The dataset was derived from three publicly available datasets and was artificially processed; therefore, no inconsistencies were present within the dataset, such as intersecting segments (Wessel and Smith 1996). As shown in Figure 5, the study data include three major New Zealand Island coastlines: South Island and North Island as the training data and Stewart Island as the test data. The starting and ending map scales of the polylines in the dataset for the simplification experiments were 1:100k and 1:250k, respectively. Polylines of 1:250k in the dataset were obtained via the DP algorithm simplification combined with manual topology correction. The raster images generated from the 1:100k polylines were used as the input data, the vertices of the 1:100k polylines were used to generate proposals, and the rectangles generated

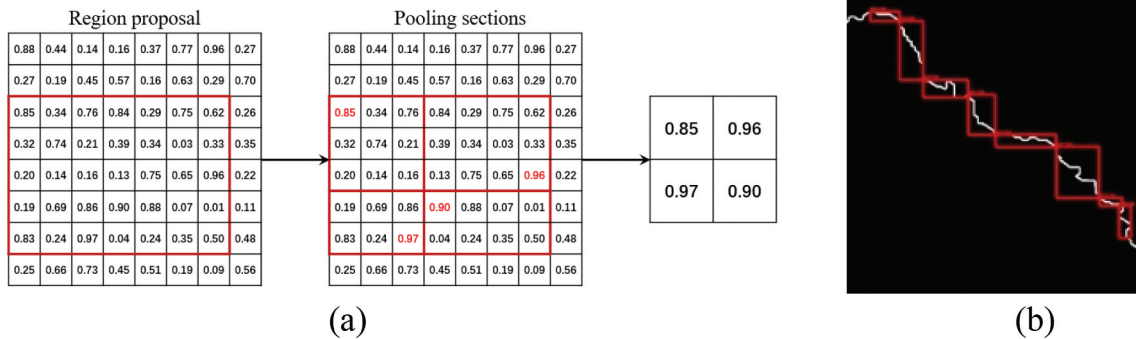
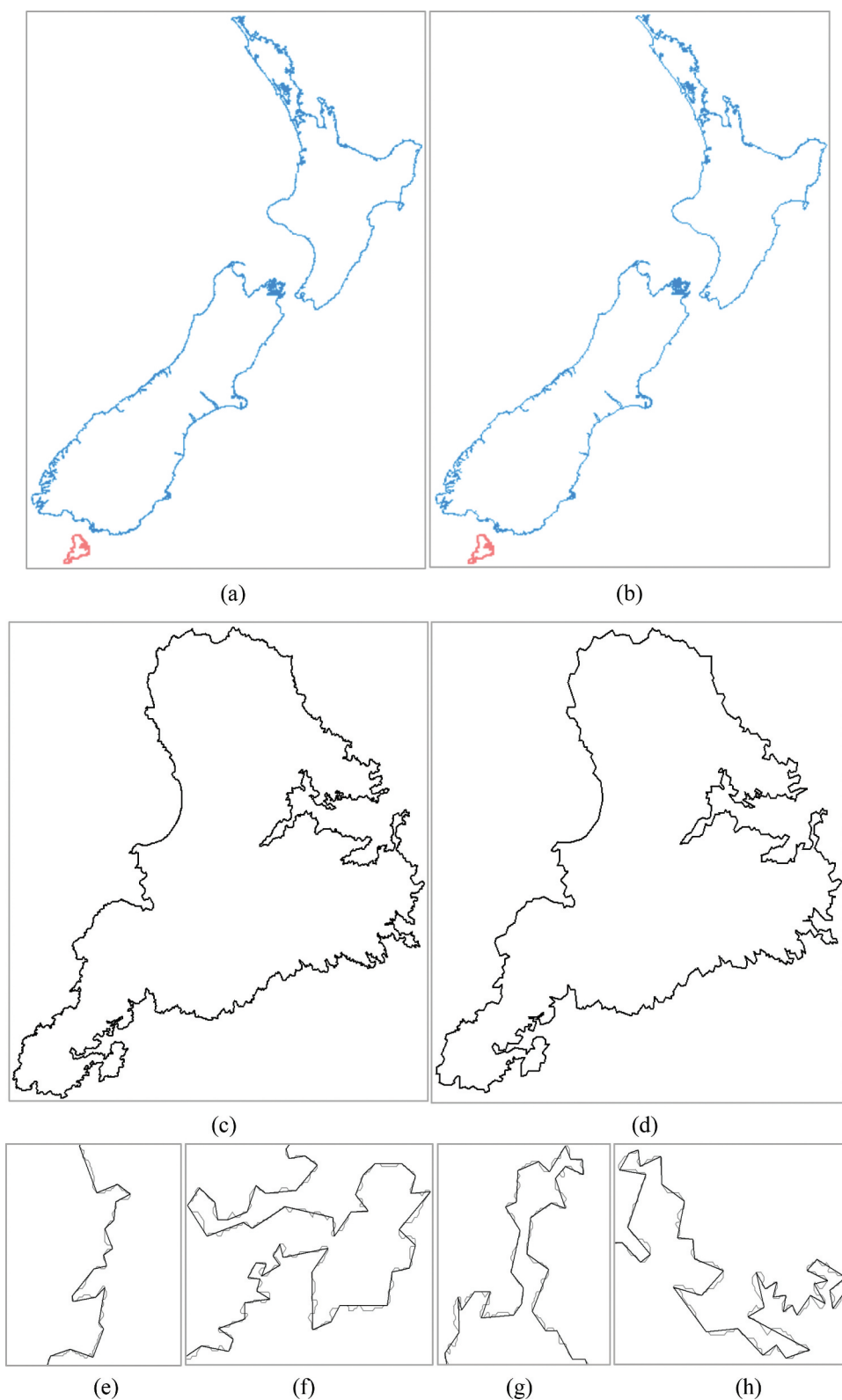


Figure 4. (a) Illustration of the ROI-pooling process. (b) red boxes represent the detected positive proposals.



**Figure 5.** Distribution of experiment data. (a) and (b) show the 1:100k and 1:250k coastlines of New Zealand's three main islands, respectively. Blue is the training data; red is the test data. (c) and (d) display the 1:100k and 1:250k Steward Island, respectively. (e)–(h) present details of coastlines before and after simplification. Gray denotes 1:100k; dark denotes 1:250k.



**Table 1.** Details of the datasets for experiments.

	Training dataset		Test dataset	
	1:100000	1:250000	1:100000	1:250000
Scale	1:100000	1:250000	1:100000	1:250000
Point number	76655	9883	3938	590
Total Length(m)	17111422	15542107	834319	720970

from every two adjacent points of the 1:250k polylines were adopted as labels to train the model. The number of vertices and total length of the training and test datasets are listed in Table 1.

#### 4.2. Experiment settings

According to the Li – Openshaw algorithm (Li and Openshaw 1992), the formula for the resolution or pixel size of the raster image is  $F_c = D/S$ . Here,  $F_c$  is the pixel size of the raster image,  $S$  is the scale of the polyline, and  $D$  is the minimum resolvable object (SVO). According to Song and Miao (2016), the SVO was set to 0.2 mm. To ensure that the details of the raster polylines are clearly visible, the pixel size  $F_c$  of the image after rasterization was set to  $F_c = D/2S$  in this study, and hence, the resolution of the image was  $F_c = \frac{1}{2} \times 0.2 \text{ mm} \times 100k = 10m$ . Additionally, to prevent sparsity in the polyline information and to avoid its effects on the accuracy of the model, the brush width, when rendering the raster polylines, was set to 5 pixels. The input image size of the model was  $1024 \times 1024$  with a window overlap rate of 50%. The model was built, trained, and tested using PyTorch (<https://pytorch.org>), and the experimental environment was a computer with two graphics processing units (NVIDIA GeForce RTX 3080).

To compare with commonly used classical methods, we selected the WM algorithm because it is also a bend-based simplification algorithm implemented in ArcGIS. It adjusts the degree of simplification by modifying the length of bend baselines, considering changes in vertex count and length. The adjusted WM parameter settings we utilized were 1500m and 650m, respectively. Additionally, given that extracting bend features from polylines requires the use of object detection techniques and fully convolutional networks like U-Net and GANs are not suitable for binary classification of bends in vector polyline simplification, we thus opted for an SVM-based polyline simplification algorithm to ensure a comprehensive comparison with other supervised methods. This algorithm begins by

generating bends. If the distance of a point from the bend baseline is greater than 70% of the bend depth, it is marked as a bend, and otherwise not. Subsequently, five bend features, including bend area, bend baseline length, bend arc length, bend depth, and bend curvature, are computed to create a feature vector. Finally, SVM is employed for vector classification, with a focus on determining the penalty coefficient (C) and kernel function. In our comparative experiments, we set C to 40, and the kernel function was specified as the Gaussian kernel function.

#### 4.3. Metrics for quantitative evaluation

The accuracy of the proposed polyline simplification method was quantitatively evaluated based on the mean average precision (mAP) metric for each pixel. The mAP value can be calculated by integrating the area below the precision – recall (P – R) curve of the bend extraction results. The precision and recall are calculated as follows:

$$\text{precision} = \frac{TP}{TP + FP}, \text{recall} = \frac{TP}{TP + FN}, \quad (2)$$

where  $TP$  represents true positive;  $FP$  represents false positive;  $TN$  represents true negative; and  $FN$  represents false negative.

The positive probability can be obtained by calculating the intersection over union (IoU):

$$\text{IoU}(A, B) = \frac{\text{area}(A \cap B)}{\text{area}(A \cup B)}, \quad (3)$$

where  $A$  is the predicted result, and  $B$  is the target label.

F1 is the harmonic mean of the precision and recall:

$$F1 = 2 * \frac{\text{precision} \times \text{recall}}{\text{precision} + \text{recall}}. \quad (4)$$

For further quantitative analysis of the simplified results, the line simplification metrics proposed by McMaster (1986) were used to evaluate the polyline

simplification results. First, the percentage change in the number of coordinates (PCNC):

$$PCNC = \left(1 - \frac{nPt_s}{nPt_o}\right) \times 100\%. \quad (5)$$

where  $nPt_s$  is the number of simplified coordinates,  $nPt_o$  is the number of original coordinates.

Second, the displacement of the total length vector (DTLV), which indicates the total “geometric displacement” of the polylines, was used to evaluate the position accuracy;

$$DTLV = \left(1 - \frac{\sum_{i=1}^{m-1} l_{s_i}}{\sum_{j=1}^{n-1} l_{s_j}}\right) \times 100\%. \quad (6)$$

where  $n$  and  $m$  refer to the number of vertices before and after simplification, respectively, and  $l_s$  refers to the length of each segment.

Third, the standard deviation of coordinates (SDC), which was used to evaluate the consistency:

$$SDC = \sqrt{\left(\sum_{i=0}^n \left(l_i - \frac{\sum_{i=0}^n l_i}{n}\right)^2\right) / n}. \quad (7)$$

where  $n$  refers to the number of vertices before simplification;  $l_i$  refers to the simplified polyline.

Additionally, according to Yu and Chen (2022), the area change ratio (Ratio\_SE) between the polylines before and after the simplification reflects the balance between shrinking and expanding areas. Ratio\_SE is calculated as follows:

$$Ratio_{SE} = \frac{|AS - AE|}{AS + AE}, \quad (8)$$

For closed polylines, the areas that exist before simplification and do not exist after simplification are

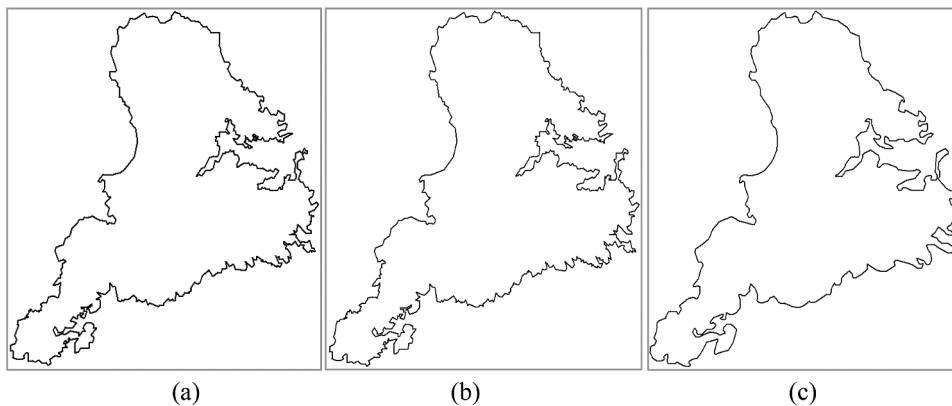
defined as shrinking areas (AS), and the areas that do not exist before simplification and exist after simplification are defined as expanding areas (AE).

#### 4.4. Experiment results

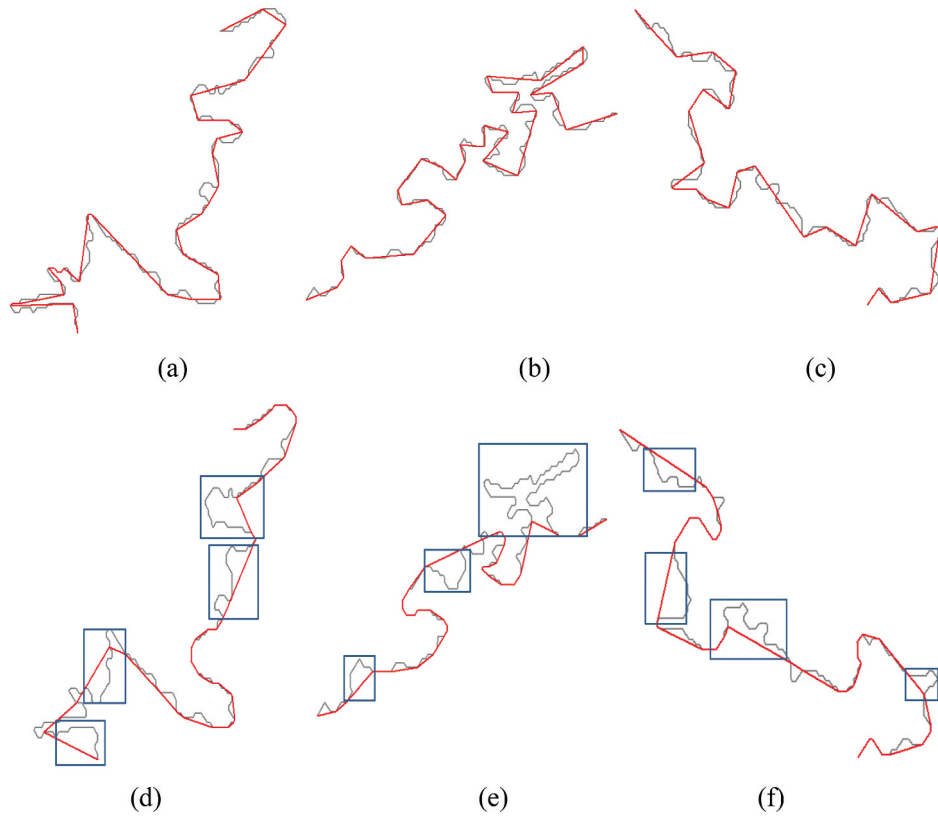
##### 4.4.1. Comparison with WM algorithm at same point change

The simplification results of the method were compared with those of the WM algorithm. First, from the perspective of data compression, the WM parameter was adjusted to ensure that a similar number of points were retained after simplification, as in the given 1:250k data. As shown in Figures 6 and 7, the simplified result of our method was highly similar to the target 1:250k coastline in terms of geometric shape features. The gently sloping shore, peninsulas sticking out from the shore, narrow bays, etc., which are difficult to handle with simplification, were simplified well by our method, which maintained the shape features of the original data. Although the result of the WM algorithm was smoother, it tended to remove the local bends from the polyline directly to avoid the problem of sharp angles, which resulted in significant deformations (blue boxes in Figure 7).

Table 2 shows the metric evaluation results for the entire coastline and three line segments, which reveal the improved performance of the proposed method. This illustrates that our method can achieve higher coordinate compression and smaller length deformation, as well as lower coordinate deviation, and has a better area-preserving ability. Figures 8 and 9 visualize the coordinate deviation and area change of the proposed method and the WM algorithm,



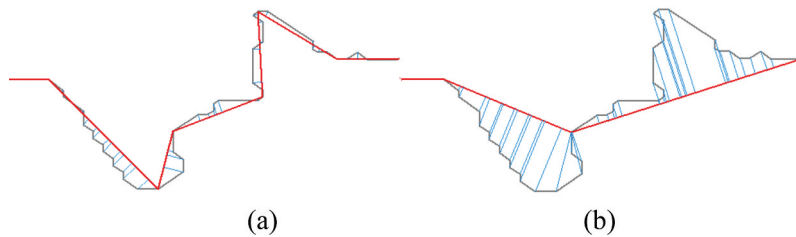
**Figure 6.** Comparisons of polyline simplification results. (a) 1:250k coastline. (b) simplified results by the proposed method. (c) simplified results by the WM algorithm.



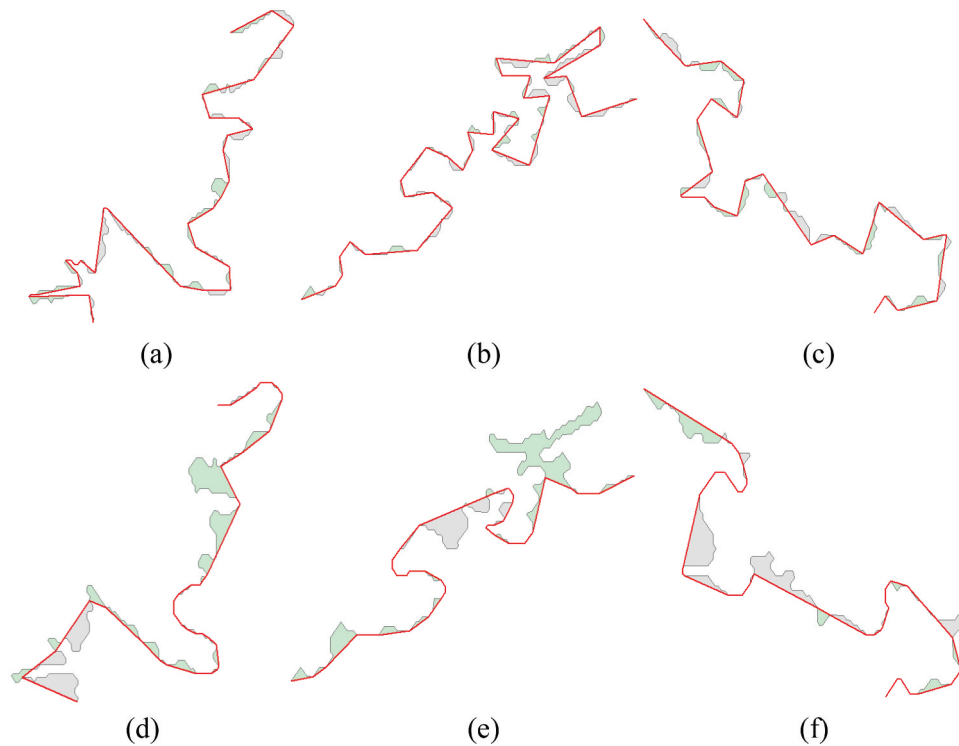
**Figure 7.** Simplified result of three different polylines (lines 1–3) obtained using the proposed method and WM algorithm. (a)–(c) simplified results obtained using the proposed method for lines 1–3, respectively. (d)–(f) simplified results obtained using the WM algorithm for lines 1–3, respectively. The blue boxes highlight the offsets of the WM algorithm.

**Table 2.** Quantitative evaluation results of the proposed method and the WM algorithm. Line1, Line2, and Line3 refer to the lines shown in Figure 7. Emboldened values represent better performance.

	Method	All	Line1	Line2	Line3
PCNC (%)	<b>Ours</b>	<b>77.99</b>	<b>80.56</b>	<b>80.36</b>	<b>80.30</b>
	WM	74.96	77.22	77.98	71.21
DTLV (%)	<b>Ours</b>	<b>13.09</b>	<b>14.81</b>	<b>14.23</b>	<b>14.20</b>
	WM	28.02	32.00	46.44	23.01
SDC (m)	<b>Ours</b>	<b>85.17</b>	<b>83.01</b>	<b>65.13</b>	<b>69.92</b>
	WM	295.73	253.91	663.04	202.22
Ratio_SE	<b>Ours</b>	<b>0.01</b>	<b>0.02</b>	<b>0.26</b>	<b>0.12</b>
	WM	0.20	0.27	0.44	0.41



**Figure 8.** Standard deviation of the coordinates. SDC results of the (a) proposed method and (b) the WM algorithm. The original polylines, simplified polylines, and distances from the original vertices to the simplified results are shown in gray, red, and blue, respectively.



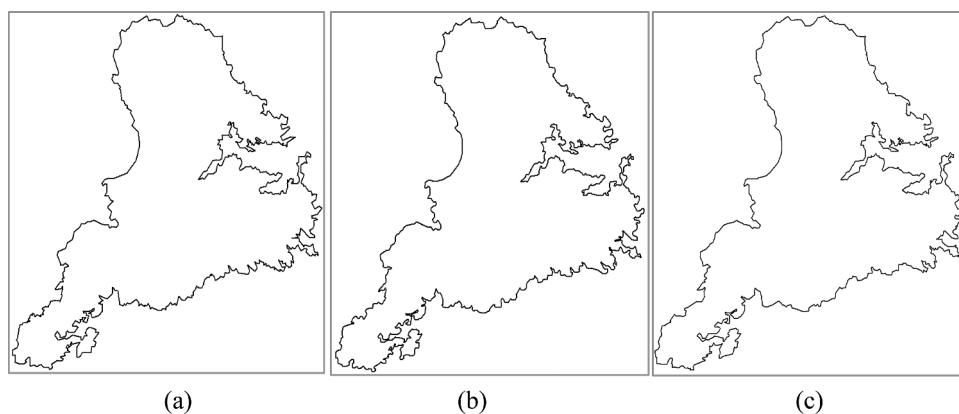
**Figure 9.** Area change of polylines before and after simplification. (a)–(c) represent the area changes generated by the proposed method of lines 1–3, respectively. (d)–(f) represent the area changes generated by the WM algorithm of lines 1–3, respectively. The shrinking areas and the expanding areas are shown in gray and green, respectively.

respectively. The simplification results of the proposed method maintained good consistency with the original polylines.

#### 4.4.2. Comparison with WM algorithm at same length change

From the perspective of shape change, the map scale can be estimated by the length of simplified lines (Ai et al. 2017). Therefore, the parameters of the WM algorithm were adjusted to retain a similar length to

that of the target 1:250k map after simplification. The simplification results are shown in Figure 10. When comparing the results of our proposed method and the SVM-based method, both methods achieved good shape preservation visually. Table 3 presents the results of a quantitative analysis of the shape indices of the three methods. Comparing the WM algorithm with the other two methods, it is evident that when the WM algorithm achieved the same length change ( $WM_L$ ), it retained too many points,



**Figure 10.** Comparisons of polyline simplification results. (a) simplified results of the proposed method. (b) simplified results of the  $WM_L$  algorithm. (c) simplified results of the SVM-based method.

**Table 3.** Quantitative evaluation results of the proposed method, the WM<sub>L</sub> algorithm and the SVM-based method.

Method	PCNC (%)	DTLV (%)	SDC (m)	Ratio_SE
<b>Ours</b>	<b>77.99</b>	13.09	<b>85.17</b>	<b>0.01</b>
WM <sub>L</sub>	52.53	<b>13.03</b>	107.26	0.19
SVM-based	72.90	19.23	138.89	<b>0.01</b>

which can cause data redundancy. Furthermore, the WM<sub>L</sub> algorithm still showed more significant coordinate displacement and area deviation in the SDC and Ratio\_SE indices, while our proposed method exhibited better shape preservation ability.

#### 4.4.3. Comparison with the SVM-based method using bend units

Recently, Duan et al. (2020) proposed a line simplification method based on SVMs using vertices and bends as local simplification units, effectively treating line simplification as a binary classification problem of either retaining vertices or bends. For the method using bends as simplification units, the authors suggested calculating five feature values for each vertex as its feature vector, followed by employing an SVM for binary classification of the vectors. The authors also introduced a generation method that involves buffer analysis of small-scale maps and large-scale maps of candidate bends, the positive – negative sample ratio for the bends in the dataset could be balanced by adjusting the buffer parameters. The simplification result is shown in Figure 10, and the quantitative evaluation of simplification result is presented in Table 3. It can be observed that the SVM-based model excels in data compression compared to WM but performs worse than our proposed method, especially in terms of DTLV and SDC. Moreover, for the SVM-based classification model, the generation of candidate bends and the setting of feature rules is closely related to expert knowledge and cartographer experience, which significantly hampers the model's generalization ability.

#### 4.4.4. Quantitative evaluation of algorithms performance

Furthermore, to evaluate the accuracy of bend recognition for both methods, a quantitative analysis was conducted based on the given 1:250k data. The proposed method automatically calculates the mAP during training to assess the performance of target recognition, achieving a mAP of 87.17%. However, as

**Table 4.** Quantitative evaluation results of the proposed method, the WM algorithm with different parameters and the SVM-based method.

Method	F1(%)	Precision (%)	Recall (%)
<b>Ours</b>	<b>79.12</b>	<b>66.51</b>	<b>97.63</b>
WM	36.72	30.74	45.59
WM <sub>L</sub>	37.28	24.48	78.13
SVM-based	71.33	59.41	89.24

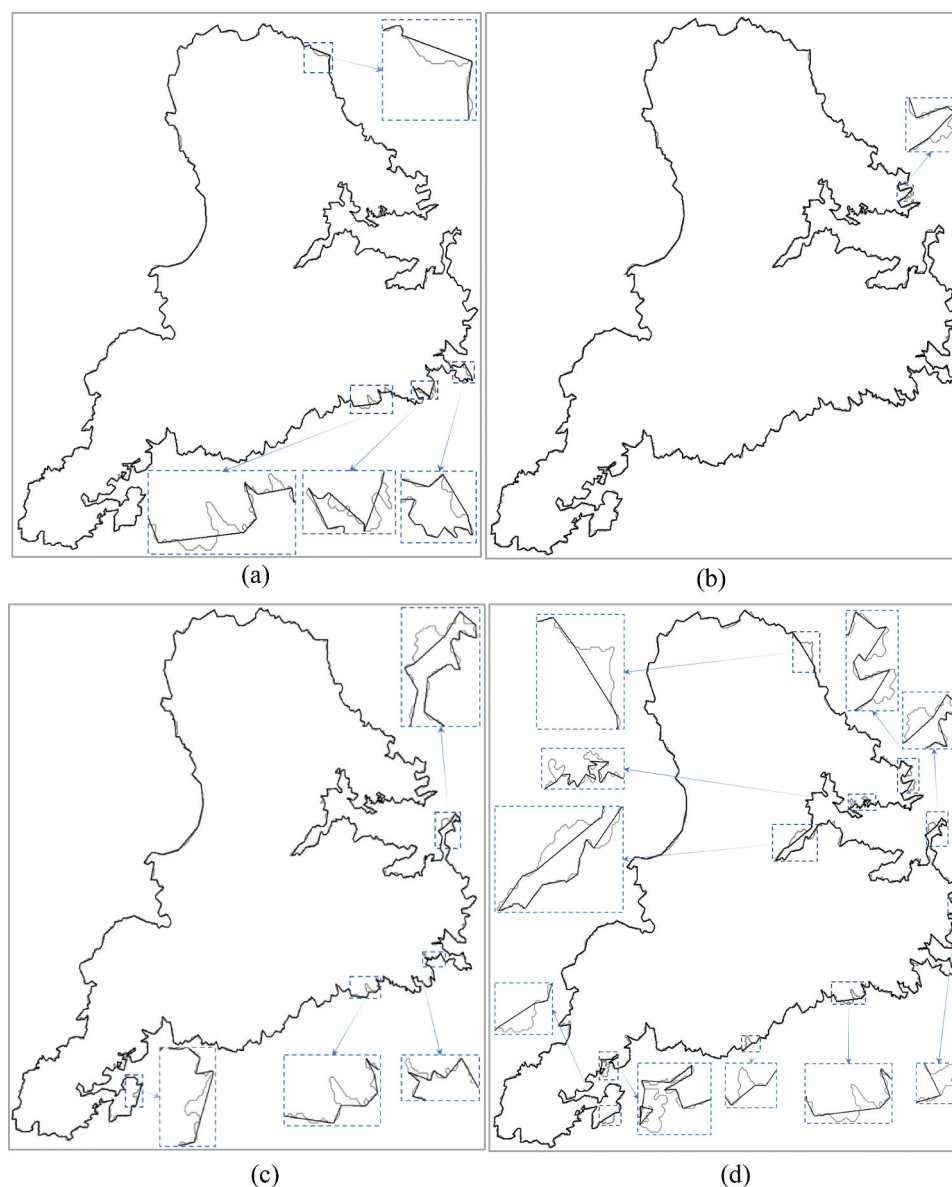
it is difficult to calculate the mAP for the WM algorithm, commonly used metrics such as F1, precision, and recall were used to evaluate its accuracy, as well as for the SVM-based method. As shown in Table 4, the quantitative evaluation results demonstrate that our proposed method outperformed the other three algorithms in all three indices. The SVM-based method is better than WM and WM<sub>L</sub>, surpassing them by more than 30% in terms of F1 but still falls short of our proposed method. WM<sub>L</sub> performs slightly better than WM, as mentioned earlier, at the cost of retaining more than 20% of the data. In conclusion, the proposed method clearly holds a significant advantage in terms of overall performance and shape preservation.

## 5. Discussion

### 5.1. Parameter sensitive analysis

#### 5.1.1. Brush width for vector data rasterization

To determine a suitable brush width for our method, the brush widths were set to 2, 5, 20, and 50 pixels for comparison and analysis. The simplification results with different brush widths at an image resolution of 10 m are shown in Figure 11, in which the blue boxes indicate the locations with distinct distortions and deviations. Evidently, the polyline with a brush width of 5 pixels shows consistent maintenance before and after simplification compared with the other three brush width results. As shown in Table 5, under the same training conditions, the three geometric evaluation metrics of DTLV, SDC, and Ratio\_SE are better than others when the brush width is 5 pixels; this indicates that the results of the



**Figure 11.** Simplification results at different brush widths. (a)  $W = 2$  pixels, (b)  $W = 5$  pixels, (c)  $W = 20$  pixels, (d)  $W = 50$  pixels. The area marked by the blue boxes is enlarged for display.

**Table 5.** Quantitative evaluation of the experimental results with different brush widths. Emboldened values represent better performance.

Brush width (pixels)	PCNC (%)	DTLV (%)	SDC (m)	Ratio_SE	mAP (%)
2	76.62	13.65	87.33	0.10	82.73
5	77.99	<b>13.09</b>	<b>85.17</b>	<b>0.01</b>	<b>87.17</b>
20	78.14	13.82	330.99	0.11	86.01
50	<b>80.68</b>	16.76	129.08	0.07	85.13

simplification, in this case, have the smallest deformation; the smallest change in length, area, and coordinate deviation; and the highest model accuracy. When the brush width is 2 pixels, the mAP value is the lowest (only 82.73%), indicating that many bends are not recognized. From the values of PCNC, DTLV,

SDC, and Ratio\_SE, although more points are retained, the shape distortion of the simplified polylines is still more noticeable than the results obtained using the 5-pixel-brush-width. When the brush width is 20 or 50 pixels, the mAP value is lower than that of 5 pixels, indicating that many bends are incorrectly detected,

**Table 6.** Quantitative evaluation of the experimental results with different input image sizes . Emboldened values represent better performance.

Image Size	PCNC (%)	DTLV (%)	SDC (m)	Ratio_SE	mAP (%)
512 × 512	72.85	13.72	289.90	0.17	78.32
1024 × 1024	<b>77.99</b>	<b>13.09</b>	<b>85.17</b>	<b>0.01</b>	<b>87.17</b>

which leads to large deformations and deviations. Through a comparative analysis of the impact on brush width settings, the model shows better learning performance when the brush width is set to 5 pixels.

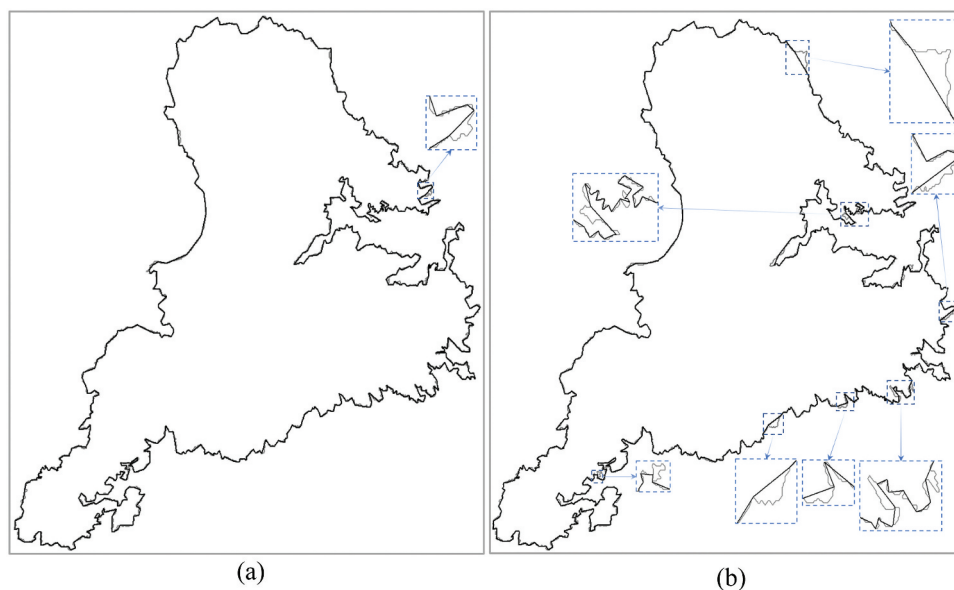
### 5.1.2. Size of input image patch

The effects of two sets of image sizes,  $512 \times 512$  and  $1024 \times 1024$ , on the model simplification performance were compared. From Table 6, it can be seen that the mAP of the model is 87.17% when the input size is  $1024 \times 1024$ , whereas the mAP is 78.32% when the input size is  $512 \times 512$ . Thus, when the input image size decreases, the continuity of the polyline is interrupted, and the positive labels of each image are reduced. This can affect the accuracy of the model in the convolutional feature extraction of the polyline and lead to many undetected bends. From the quantitative analysis results – although more points were retained when the input image size was set to  $512 \times 512$ —the total length deformation and coordinate deviation of the simplification results were larger than those corresponding to the  $1024 \times 1024$  input. Figure 12 shows the simplification results for different

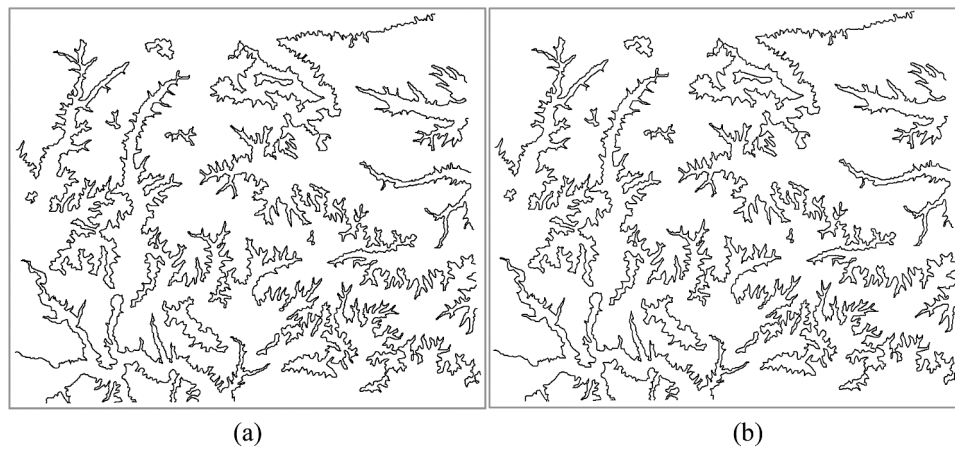
input image sizes, where the blue boxes indicate evident deformations. It can be observed that an input size of  $1024 \times 1024$  can better preserve geometric features and shows better simplification results.

### 5.2. Robustness analysis

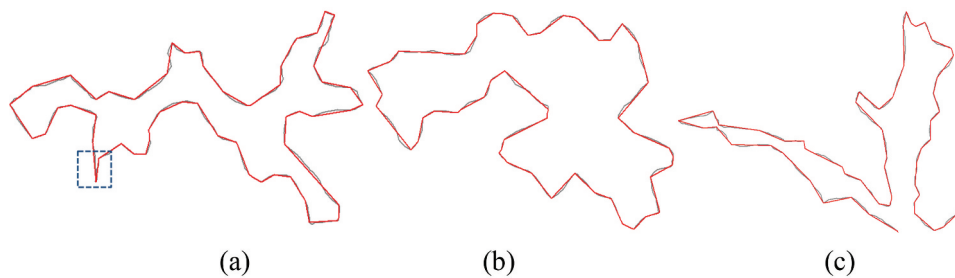
The simplification rules of different types of polylines differ greatly. To verify the generalization effect of the proposed method on other types of polylines, contour data were obtained from the Geospatial Data Cloud platform (<http://www.gscloud.cn/search>) for simplification experiments. The scales of the data before and after the simplification were 1:100k and 1:250k, respectively. The experimental results show that the proposed method achieved a coordinate point compression rate of 81.76%; the simplification result of the total data and three different contour lines is shown in Figures 13 and 14. The proposed method retained good shape features of the contour lines. In addition, it is worth noting that the training labels of the proposed method were generated



**Figure 12.** Simplification results with different input image sizes: (a)  $1024 \times 1024$  and (b)  $512 \times 512$ . The area marked by the blue boxes is enlarged for display.



**Figure 13.** Simplification results of contours. (a) Source contours. (b) Simplified results.



**Figure 14.** Simplification results details. (a)–(c) Different contours simplified by our method. The blue boxes indicate the sharp angles that appear after simplification.

according to the DP algorithm. Thus, when the knowledge learned by the model is applied to line simplification, the problem of sharp corners persists, similar to that in the DP algorithm; nonetheless, this demonstrates that the proposed method can effectively learn the simplification rules embedded in the existing simplification cases.

### 5.3. Limitations

This paper introduces a new concept of deep-learning-based target detection to detect line bends for automatic vector polyline simplification. However, as shown in Figure 14(a), when the labeled data have sharp angles, the performance of the proposed method is compromised. Although the problem can be solved by finding a more suitable training dataset, topological errors in the simplification results cannot be completely avoided.

To further improve the performance of the proposed method under various scenarios, effective strategies for handling sharp angles and topological errors should be explored. A possible solution is to

integrate the generation of GANs into the simplification process. The generation of new points in the necessary sections may solve problems such as sharpness and topological intersections during simplification.

Additionally, remote sensing image mapping is rich in texture and color features, which can better leverage the image feature extraction capability of convolutional networks – if our method is used to simplify vector polylines extracted from remote sensing images based on boundary tracking, it can better facilitate line simplification and enable the implementation of automatic mapping of images to vectors.

## 6. Conclusions

Automatic simplification of vector polylines is crucial in map generalization. In this paper, we proposed an automatic polyline simplification method based on an RPN that can be trained end-to-end and integrate raster and vector features to achieve effective vector-to-vector simplification. The simplified results of the coastline show that the proposed method can learn



simplification knowledge and potential rules from existing simplification cases without manual intervention, and leverage the vector and raster features of polylines to achieve automatic bend detection, as well as realize end-to-end and vector-to-vector polyline simplification. The experiments also indicate that the simplified results have a higher compression ratio of coordinate points and lower shape deformation and deviation than the results of the traditional WM and SVM-based algorithms.

Additionally, considering that there are many available remote sensing image datasets, it is expected that the proposed method can be applied to support the automatic mapping of images to vectors, such as shoreline detection. Nevertheless, our proposed method still has limitations, such as producing bend exaggeration, and thus can be further improved in the future to enable wider application in the automatic generalization of different geographical objects.

### Disclosure statement

No potential conflict of interest was reported by the author(s).

### Funding

This work was supported by the Key Laboratory of Geological Survey and Evaluation of Ministry of Education [GLAB2020ZR11]; National Natural Science Foundation of China [42171408].

### ORCID

Shaofen Xu  <http://orcid.org/0000-0002-9671-7430>

### Data availability statement

The experimental data used in this study are available in the coastline dataset GSHHG (Wessel and Smith 1996) (<http://www.soest.hawaii.edu/wessel/gshhg>).

### References

- Ai, T. 2021. "Some Thoughts on Deep Learning Enabling Cartography." *Acta Geodaetica et Cartographica Sinica* 50 (9): 1170. <https://doi.org/10.11947/j.AGCS.2021.20210091>.
- Ai, T., S. Ke, M. Yang, and J. Li. 2017. "Envelope Generation and Simplification of Polyline Using Delaunay Triangulation." *International Journal of Geographical Information Science* 31 (2): 297–319. <https://doi.org/10.1080/13658816.2016.1197399>.
- Cheng, B., Q. Liu, X. Li, and Y. Wang. 2013. "Building Simplification Using Backpropagation Neural Networks: A Combination of Cartographers' Expertise and Raster-Based Local Perception." *GIScience & Remote Sensing* 50 (5): 527–542. <https://doi.org/10.1080/15481603.2013.823748>.
- Chen, Q., L. Wang, S. L. Waslander, and X. Liu. 2020. "An End-To-End Shape Modeling Framework for Vectorized Building Outline Generation from Aerial Images." *ISPRS Journal of Photogrammetry and Remote Sensing* 170 (December): 114–126. <https://doi.org/10.1016/j.isprsjprs.2020.10.008>.
- Courtial, A., A. E. Ayedi, G. Touya, and X. Zhang. 2020. "Exploring the Potential of Deep Learning Segmentation for Mountain Roads Generalisation." *ISPRS International Journal of Geo-Information* 9 (5): 338. <https://doi.org/10.3390/ijgi9050338>.
- Courtial, A., G. Touya, and X. Zhang. 2021. "Generative Adversarial Networks to Generalise Urban Areas in Topographic Maps." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLIII-B4-2021* (June): 15–22. <https://doi.org/10.5194/isprs-archives-XLIII-B4-2021-15-2021>.
- Courtial, A., G. Touya, and X. Zhang. 2022. "Constraint-Based Evaluation of Map Images Generalized by Deep Learning." *Journal of Geovisualization and Spatial Analysis* 6 (1): 13. <https://doi.org/10.1007/s41651-022-00104-2>.
- Courtial, A., G. Touya, and X. Zhang. 2023. "Deriving Map Images of Generalised Mountain Roads with Generative Adversarial Networks." *International Journal of Geographical Information Science* 37 (3): 499–528. <https://doi.org/10.1080/13658816.2022.2123488>.
- Douglas, D. H., and T. K. Peucker. 1973. "Algorithms for the Reduction of the Number of Points Required to Represent a Digitized Line or Its Caricature." *Cartographica: The International Journal for Geographic Information and Geovisualization* 10 (2): 112–122. <https://doi.org/10.3138/FM57-6770-U75U-7727>.
- Duan, P., H. Qian, H. He, L. Xie, and D. Luo. 2020. "A Line Simplification Method Based on Support Vector Machine." *Geomatics and Information Science of Wuhan University* 45 (5): 744–52, 783. <https://doi.org/10.13203/j.whugis20180434>.
- Du, J., F. Wu, Z. H. U. Li, C. Liu, and A. Wang. 2022. "An Ensemble Learning Simplification Approach Based on Multiple Machine-Learning Algorithms with the Fusion Using of Raster and Vector Data and a Use Case of Coastline Simplification." *Acta Geodaetica et Cartographica Sinica* 51 (3): 373. <https://doi.org/10.11947/j.AGCS.2022.20210135>.
- Du, J., F. Wu, R. Xing, X. Gong, and L. Yu. 2021. "Segmentation and Sampling Method for Complex Polyline Generalization Based on a Generative Adversarial Network." *Geocarto International* 37 (14), February): 4158–4180. <https://doi.org/10.1080/10106049.2021.1878288>.
- Feng, Y., F. Thiemann, and M. Sester. 2019. "Learning Cartographic Building Generalization with Deep

- Convolutional Neural Networks." *ISPRS International Journal of Geo-Information* 8 (6): 258. <https://doi.org/10.3390/ijgi8060258>.
- Girshick, R. 2015. "Fast R-CNN." In *2015 IEEE International Conference on Computer Vision (ICCV)* 1440–1448. <https://doi.org/10.1109/ICCV.2015.169>.
- Isola, P., J. Y. Zhu, T. Zhou, and A. A. Efros. 2017. "Image-To-Image Translation with Conditional Adversarial Networks." In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)* 5967–5976. Honolulu, HI: IEEE. Accessed August 23, 2022. <https://doi.org/10.1109/CVPR.2017.632>.
- Jiang, B. 2003. "Line Simplification Using Self Organizing Maps," January. Accessed November 25, 2022. <https://www.academia.edu/2611465>.
- Kang, Y., S. Gao, and R. E. Roth. 2019. "Transferring Multiscale Map Styles Using Generative Adversarial Networks." *International Journal of Cartography* 5 (2–3): 115–141. <https://doi.org/10.1080/23729333.2019.1615729>.
- Karsznia, I., and K. Sielicka. 2020. "When Traditional Selection Fails: How to Improve Settlement Selection for Small-Scale Maps Using Machine Learning." *ISPRS International Journal of Geo-Information* 9 (4): 230. <https://doi.org/10.3390/ijgi9040230>.
- Karsznia, I., and R. Weibel. 2018. "Improving Settlement Selection for Small-Scale Maps Using Data Enrichment and Machine Learning." *Cartography and Geographic Information Science* 45 (2): 111–127. <https://doi.org/10.1080/15230406.2016.1274237>.
- Kent, A. 2017. "Trust Me, I'm a Cartographer: Post-Truth and the Problem of Acritical Cartography." *The Cartographic Journal* 54 (3): 193–195. <https://doi.org/10.1080/00087041.2017.1376489>.
- LeCun, Y., Y. Bengio, and G. Hinton. 2015. "Deep Learning." *Nature* 521 (7553): 436–444. <https://doi.org/10.1038/nature14539>.
- Lee, J., H. Jang, J. Yang, and K. Yu. 2017. "Machine Learning Classification of Buildings for Map Generalization." *ISPRS International Journal of Geo-Information* 6 (10): 309. <https://doi.org/10.3390/ijgi6100309>.
- Li, W., and C.-Y. Hsu. 2020. "Automated Terrain Feature Identification from Remote Sensing Imagery: A Deep Learning Approach." *International Journal of Geographical Information Science* 34 (4): 637–660. <https://doi.org/10.1080/13658816.2018.1542697>.
- Li, Z., and S. Openshaw. 1992. "Algorithms for Automated Line Generalization 1 Based on a Natural Principle of Objective Generalization." *International Journal of Geographical Information Systems* 6 (5): 373–389. <https://doi.org/10.1080/02693799208901921>.
- Liu, P., X. Li, W. Liu, and T. Ai. 2016. "Fourier-Based Multi-Scale Representation and Progressive Transmission of Cartographic Curves on the Internet." *Cartography and Geographic Information Science* 43 (5): 454–468. <https://doi.org/10.1080/15230406.2015.1088799>.
- McMaster, R. B. 1986. "A Statistical Analysis of Mathematical Measures for Linear Simplification." *The American Cartographer* 13 (2): 103–116. <https://doi.org/10.1559/152304086783900059>.
- Park, W., and K. Yu. 2010. "Generalization of Digital Topographic Map Using Hybrid Line Simplification." In *Proceedings of the ASPRS 2010 Annual Conference* 6. San Diego.
- Rao, J., S. Gao, Y. Kang, and Q. Huang. 2020. "LSTM-Trajgan: A Deep Learning Approach to Trajectory Privacy Protection." *arXiv Accessed*. <https://doi.org/10.48550/arXiv.2006.10521>. November 25, 2022.
- Ren, S., K. He, R. Girshick, and J. Sun. 2017. "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks." *IEEE Transactions on Pattern Analysis and Machine Intelligence* 39 (6): 1137–1149. <https://doi.org/10.1109/TPAMI.2016.2577031>.
- Ronneberger, O., P. Fischer, and T. Brox. 2015. "U-Net: Convolutional Networks for Biomedical Image Segmentation." In *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2015*, edited by N. Navab, J. Hornegger, M. W. William, and F. F. Alejandro, Lecture Notes in Computer Science 234–241. Cham: Springer International Publishing. [https://doi.org/10.1007/978-3-319-24574-4\\_28](https://doi.org/10.1007/978-3-319-24574-4_28).
- Sandler, M., A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen. 2018. "Mobilenetv2: Inverted Residuals and Linear Bottlenecks." In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition* (Salt Lake City, UT, USA) 4510–4520 doi:10.1109/CVPR.2018.00474.
- Sester, M., Y. Feng, and F. Thiemann. 2018. "BUILDING GENERALIZATION USING DEEP LEARNING." *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences XLII-4* (September): 565–572. <https://doi.org/10.5194/isprs-archives-XLII-4-565-2018>.
- Song, J., and R. Miao. 2016. "A Novel Evaluation Approach for Line Simplification Algorithms Towards Vector Map Visualization." *ISPRS International Journal of Geo-Information* 5 (12): 223. <https://doi.org/10.3390/ijgi5120223>.
- Stanislawski, L. V., B. P. Buttenfield, P. Bereuter, S. Savino, and C. A. Brewer. 2014. "Generalisation Operators". In *Abstracting Geographic Information in a Data Rich World*, In edited by D. Burghardt, C. Duchêne, and W. Mackaness, Lecture Notes in Geoinformation and Cartography 157–195. Cham: Springer International Publishing. Accessed November 25, 2022. [https://doi.org/10.1007/978-3-319-00203-3\\_6](https://doi.org/10.1007/978-3-319-00203-3_6)
- Touya, G., X. Zhang, and I. Lokhat. 2019. "Is Deep Learning the New Agent for Map Generalization?" *International Journal of Cartography* 5 (2–3): 142–157. <https://doi.org/10.1080/23729333.2019.1613071>.
- Visvalingam, M., and J. D. Whyatt. 1993. "Line Generalisation by Repeated Elimination of Points." *The Cartographic Journal* 30 (1): 46–51. <https://doi.org/10.1179/caj.1993.30.1.46>.
- Vrotsou, K., H. Janetzko, C. Navarra, G. Fuchs, D. Spretke, F. Mansmann, N. Andrienko, and G. Andrienko. 2015. "SimpliFLy: A Methodology for Simplification and Thematic

- Enhancement of Trajectories." *IEEE Transactions on Visualization and Computer Graphics* 21 (1): 107–121. <https://doi.org/10.1109/TVCG.2014.2337333>.
- Wang, Z., and J.-C. Müller. 1998. "Line Generalization Based on Analysis of Shape Characteristics." *Cartography and Geographic Information Systems* 25 (1): 3–15. <https://doi.org/10.1559/152304098782441750>.
- Wessel, P., and W. H. F. Smith. 1996. "A Global, Self-Consistent, Hierarchical, High-Resolution Shoreline Database." *Journal of Geophysical Research: Solid Earth* 101 (B4): 8741–8743. <https://doi.org/10.1029/96JB00104>.
- Yan, X., T. Ai, M. Yang, X. Tong, and Q. Liu. 2020. "A Graph Deep Learning Approach for Urban Building Grouping." *Geocarto International*: 1–24. <https://doi.org/10.1080/10106049.2020.1856195>.
- Yang, M., T. Yuan, X. Yan, T. Ai, and C. Jiang. 2022. "A Hybrid Approach to Building Simplification with an Evaluator from a Backpropagation Neural Network." *International Journal of Geographical Information Science* 36 (2): 280–309. <https://doi.org/10.1080/13658816.2021.1873998>.
- Yu, W., and Y. Chen. 2022. "Data-Driven Polyline Simplification Using a Stacked Autoencoder-Based Deep Neural Network." *Transactions in GIS* 26 (5): 2302–2325. June, tgis.12965. <https://doi.org/10.1111/tgis.12965>.
- Zhou, Q., and Z. Li. 2017. "A Comparative Study of Various Supervised Learning Approaches to Selective Omission in a Road Network." *The Cartographic Journal* 54 (3): 254–264. <https://doi.org/10.1179/1743277414Y.0000000083>.