UTS UNIVERSITY OF TECHNOLOGY SYDNEY

ARBITRARY-SHAPE SCENE TEXT DETECTION AND ITS APPLICATION IN EDUCATIONAL RESOURCE NAVIGATION

by Chengpei Xu

Thesis submitted in fulfilment of the requirements for
the degree of

Doctor of Philosophy

under the supervision of Prof. Sean He and Dr. Wenjing Jia

University of Technology Sydney
Faculty of Engineering and Information Technology

June 2022

Certificate of Authorship/Originality

I, Chengpei Xu, declare that this thesis is submitted in fulfilment of the requirements for the award of Doctor of Philosophy, in the School of Electrical and Data Engineering, Faculty of Engineering and Information Technology, at the University of Technology Sydney.

This thesis is wholly my own work unless otherwise referenced or acknowledged. In addition, I certify that all information sources and literature used are indicated in the thesis.

This document has not been submitted for qualifications at any other academic institution.

This research is supported by the Australian Government Research Training Program and the UTS FEIT Research Scholarship.

Signature: Production Note:
Signature removed prior to publication.

Date: June 22, 2022

Acknowledgements

I wish to express my deepest appreciation to my supervisor, Professor Xiangjian He, for his excellent supervision, invaluable advice, and encouragement during my doctoral study. I also wish to express my deepest thanks to Professor He for trusting me and providing me an opportunity to get UTS FEIT Research Scholarship. From the bottom of my heart I know, I can always get help and suggestion from him during my downtime in life and study. It is so true.

I also express my sincere gratitude to my co-supervisor Dr. Wenjing Jia. I still remember how confused I was when I first came to UTS for finding the right research direction. Wenjing's hands-on guidance and insightful analysis helped me to find the right research that I was interested in and good at, making my research journey not boring. I admire Dr. Wenjing Jia's professionalism as a teacher, as I cannot remember how many times Wenjing and I have discussed research issues late into the night.

I also wish to express special thanks to Professor Ruomei Wang, Professor Xiaonan Luo, Professor Zhongxuan Luo, Professor Zhixun Su, Dr. Baoquan Zhao, Dr. Boliang Guan, Dr. Hanhui Li, Dr. Xiaochen Fan, Dr. Qingqing Wang, Mr. Tingcheng Cui, Mr. Lijie Shao, Miss Mengqiu Hu, Mr. Jiachen Kang and Mr. YuanFang Zhang for all their help and valuable discussions.

Most importantly I would like to thank my parents for their years of support. Due to COVID-19, I have not seen them since mid 2019. I have left my hometown Xinjiang for higher education in 2010, four years in Changsha, one year in Guangzhou and seven years in Sydney. I miss them and hope they are proud of what I have achieved today. Finally, I am deeply grateful to my wife Si Wang. I would not have been able to complete my doctorate degree without her dedication.

List of Publications

This thesis is based on the following publications:

- **Chengpei Xu**, Wenjing Jia, Tingcheng Cui, Ruomei Wang, Yuan-fang Zhang, Xiangjian He, Arbitrary-shape Scene Text Detection via Visual-Relational Rectification and Contour Approximation, 'IEEE Transactions on Multimedia, DOI: 10.1109/TMM.2022.3171085.' (Accepted; covering most parts of Chapter 3)

- **Chengpei Xu**, Wenjing Jia, Ruomei Wang, Xiaonan Luo, Xiangjian He, MorphText: Deep Morphology Regularized Accurate Arbitrary-shape Scene Text Detection, 'IEEE Transactions on Multimedia, DOI: 10.1109/TMM.2022.3172547' (Accepted; covering most part of Chapter 4)

- **Chengpei Xu**, Ruomei Wang, Shujin Lin, Xiaonan Luo, Boquan Zhao, Lijie Shao, Mengqiu Hu, Lecture2Note: Automatic Generation of Lecture Notes from Slide-Based Educational Videos. *in* 'Proceeding of IEEE International Conference on Multimedia and Expo (ICME), Shanghai, 2019, pp. 898-903' (covering some parts of Chapter 5)

- **Chengpei Xu**, Wenjing Jia, Si Wang, Ruomei Wang, Xiangjian He, Shujin Lin, Baoquan Zhao, Yuan-fang Zhang, Semantic Navigation of Slide-based Educational Video for AutoNote Generation, 'IEEE Transactions on Learning Technologies' (Under review after major revision; covering most parts of Chapter 5)

Contents

List of Figures

List of Tables

Abbreviation

AAAI AAAI Conference on Artificial Intelligence
Adam - Adaptive Moment Estimation
AP - Affinity Propagation
CNN - Convolutional Neural Network
CVPR International Conference on Computer Vision and Pattern Recognition
DMCL - Deep Morphological Closing
DMN - Deep Morphological Network
DMOP - Deep Morphological Opening
ECCV European Conference on Computer Vision
EMD - Earth Mover's Distance
FCN - Fully Convolutional Network
FD - Fuse Decoding
FPN - Feature Pyramid Network
FPNS - False Positive/Negative Suppression
GCN - Graph Convolutional Neural Network
GGTR - Graph Guided Text Region
GPU - Graphical Processing Unit
ICCV International Conference on Computer Vision Recognition
IJCAI The International Joint Conference on Artificial Intelligence
IoU - Intersection over Union
LAT - Location-Aware Transfer
MLT Multi Lingual Text
MM ACM International Conference on Multimedia
MOOC - Massive Open Online Courses
MSRA-TD Microsoft Research Asia Text Detection
NMS - Non-maximum Suppression
OCR - Optical Character Recognition
OER - Open Educational Resources
PR Pattern Recognition
ReLU - Rectified Linear Unit
RNN - Recurrent Neural Network
SAp - Shape Approximation
SE - Structure Element
SGD - Stochastic Gradient Descent
SOTA - State of The Art
TCL - Text Center Line
TIP Transaction on Image Processing
TMM Transaction on Multimedia
TOC - Table of Content
TR - Text Region
WMD - Word Mover's Distance

Dedication

To my wife Si Wang

To my parents Yan Li and Bin Xu

ABSTRACT

ARBITRARY-SHAPE SCENE TEXT DETECTION AND ITS APPLICATION IN EDUCATIONAL RESOURCE NAVIGATION

by

Chengpei Xu

Text instances exist widely as an information carrier in natural scenes, videos and document photos. However, localizing text instances with arbitrary shapes is a challenging task since their style, colour, size, aspect ratio and shape vary greatly depending on the using scenarios. The abovementioned issues hinder the retrieval of information and the digitization of raw photos and videos. The situation worsens when the raw photos and videos are for educational purposes.

In this thesis, we address the challenging problem of arbitrary-shape scene text detection by proposing two deep learning-based bottom-up approaches. Then, we create a navigation system for slide-based educational resources using the semantic information of the detected texts as the primary cue.

In the first approach, we revitalize the GCN-based bottom-up text detection frameworks by aggregating the visual-relational features of text with two effective false positive/negative suppression mechanisms. First, dense overlapping text segments depicting the "characterness" and "streamline" of text are generated for further relational reasoning and weakly supervised segment classification. Then, a Location-Aware Transfer (LAT) module is designed to transfer text's relational features into visual compatible features with a Fuse Decoding (FD) module to enhance the representation of text regions for the second step suppression. Finally, a novel multiple-text-map-aware contour-approximation strategy is developed, instead of the route-finding process.

In the second approach, targeting building reliable connections between text

segments and alleviating error accumulation in bottom-up modelling, we propose a novel approach to capture the regularity of texts by embedding deep morphology for arbitrary-shape text detection so as to regularize false text segment detection and link missing connections. Towards this end, two deep morphological modules are designed to regularize text segments and determine the linkage between them. First, a Deep Morphological Opening (DMOP) module is constructed to remove false text segment detection accumulated in the feature extraction process. Then, a Deep Morphological Closing (DMCL) module is proposed to allow text instances of various shapes to stretch their morphology in all directions while deriving their connections.

Using the detected arbitrary-shape text information in educational resources as a primary cue, we propose a slide-based video navigation tool that can extract the hierarchical structure and semantic relationship of visual entries in videos by integrating multi-channel information. A clustering approach is proposed for restoring the hierarchical relationship between visual entities. The restored visual entities are then associated with their corresponding audio speech text by evaluating their semantic relationship.

Dissertation directed by Professor Xiangjian (Sean) He
Dissertation co-directed by Dr. Wenjing Jia
Faculty of Engineering and Information Technology
University of Technology Sydney

Chapter 1

Introduction

1.1 Introduction

Scene Text detection is the upstream task of many downstream tasks such as text recognition [3], document visual question answering [4], image retrieval [5], autonomous driving [6], layout analyse [7], video content navigation [8] etc. With the popularity of smartphones and mobile cameras, individuals can take pictures and videos from any time and anywhere, leading to a dramatic increase in videos and images containing text instances. At the same time, with the rapid development of open educational resources (OER), a considerable number of online educational resources have emerged on various MOOC platforms such as Coursera*, Udacity† and YouTube‡. In addition to the elaborate courses by some prestigious universities, there are also many homemade instructional courses. These educational resources usually contain highlighted and summary knowledge points (text instances).

However, text detection is still a challenging task since the orientation, colour, size, aspect ratio and shape of text change frequently depending on the scenario and purpose of usage. Text instances are embedded in complex backgrounds, e.g., advertisements, traffic signs, packaging and posters, and the scenarios can range from natural scenes to classrooms. Text instances can be horizontal or vertical, curved, angled, and they can be of arbitrary shape depending on the scenarios. In addition to the complexity and variety of scenarios in which text instances exist, the manner and quality in which they are recorded make it more challenging to be localized. For example, the imperfect shooting condition of videos and photos also

*https://www.coursera.org/

†https://www.udacity.com/

‡https://www.youtube.com/

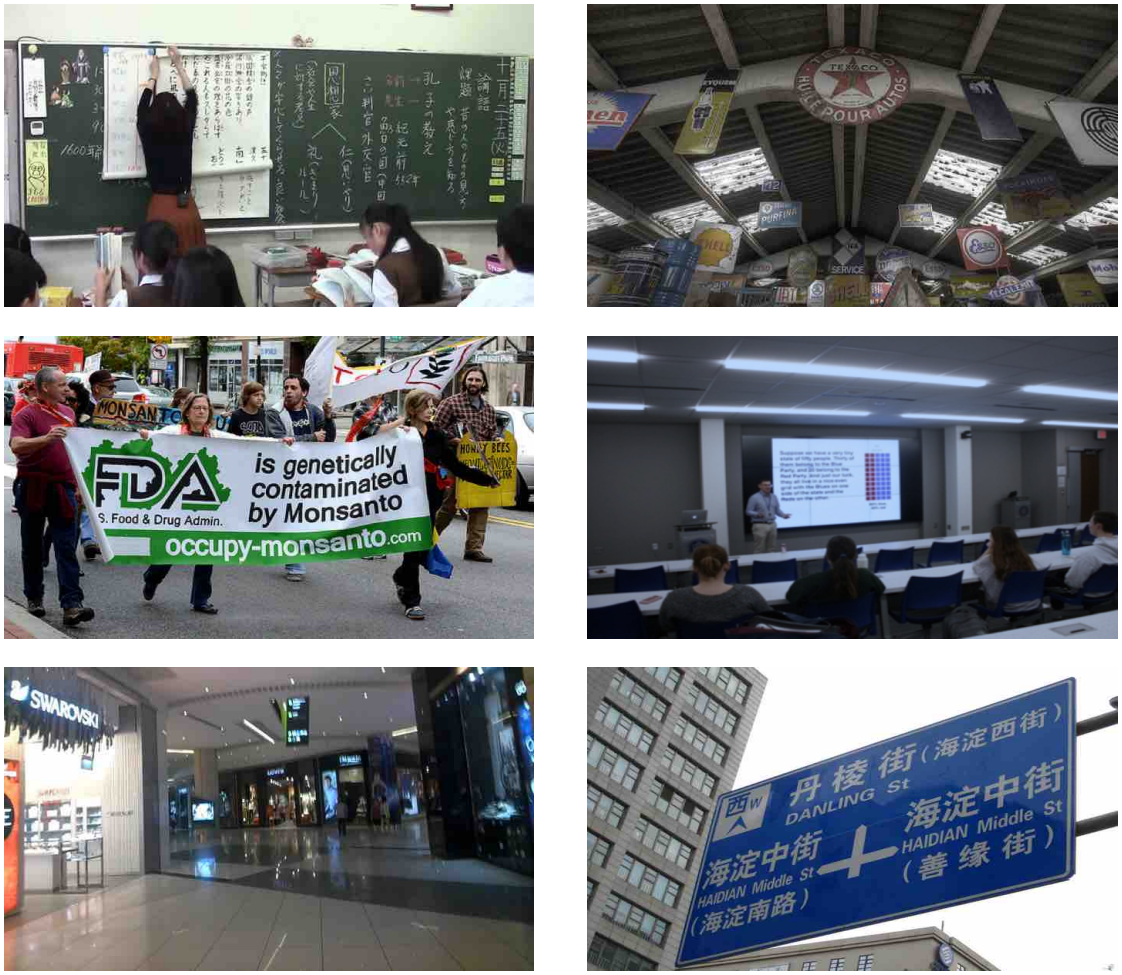


Figure 1.1 : The illustration of various arbitrary shape text instances in complex environments and discretionary shooting conditions.

affects the appearance of text instances, which may lead to text instances being deformed, blurred, and uneven exposure. These arbitrarily shaped texts in complex environments and shooting conditions greatly increase the difficulty of locating text areas. Especially for the educational scene, a professional camera is not always available in education classes to help students record high-quality course videos and photos. Thus, for some courses without recording, the only photos of key knowledge points are taken by some portable devices in a discretionary shooting condition. Figure 1.1 illustrates various and arbitrary-shaped text instances in a complex environment and discretionary shooting conditions.

For more accurate localization of arbitrary-shaped text instances, some methods

(a) Contours of quadrilateral (b) Contours of arbitrary-shape

Figure 1.2 : Different annotation methods of multi-oriented texts and curved texts

in the research community [9–11] replaced the quadrilateral labelling to reduce the interference of background noise in the text instances and created a more accurate contour representation (see Figure 1.2 for example). Detecting text instances using curved contour has become the primary research trend since it can provide a convenient representation of text instances of any shape and angle. The accuracy of arbitrary shape scene text detection also directly or indirectly affects the performance of downstream tasks. Thus, accurately localizing arbitrary shape text instances in complex environments and discretionary shooting conditions is an essential step for retrieving and digitizing the information in text instances. For example, the idea of creating navigation tools for educational videos to improve learners' learning efficiency can only be achieved under the condition that text instances are accurately detected. From the practical perspective, however, as an application of arbitrary-shape scene text detection, the navigation tool for educational videos that can only roughly locate the text instances is not enough for the purpose of learning due to that the text instances are not being integrated into a complete knowledge point and not linked with its explanation.

This thesis, therefore, focuses on the challenges related to arbitrary-shaped scene text detection. In addition to the downstream tasks of arbitrary-shaped scene text detection, we propose a novel navigation tool for educational videos using the detected text instances as the primary cue.

(a) A bottom-up modelling example from [22]

(b) A top-down modelling example from [23]

Figure 1.3 : Different modelling examples

1.2 Issues and Challenges

1.2.1 Arbitrary-shape Scene Text Detection

There are two modelling methods for arbitrary-shape scene text detection, *i.e.*, the bottom-up modelling methods and the top-down modelling methods, as shown in Figure 1.3. Most of the top-down modelling methods globally model the contour of text areas in a single pass. Thus, methods such as [9, 10, 12] which directly or indirectly adopt the regression idea to predict the points on the contour can be grouped into top-down modelling methods. Moreover, those methods such as [2, 13–16] producing text areas directly from the segmentation mask can also be grouped into top-down modelling methods.

Contrary to the top-down modelling methods, the bottom-up arbitrary-shape scene text detection methods [11, 17–21] locally model text areas into text segments and group them by judging their linkage relationship.

Top-down methods draw lessons from some successful modules designed for object detection [24–26] and semantic/instance segmentation [16, 27] can be directly embedded into top-down methods to enhance the detection performance. However, texts differ significantly from general objects in terms of scale, aspect ratio, etc. Some direct embedding operations have not considered the characters of the text themselves. As top-down methods consider globally generating the final contour of text in one go, the size of the CNN's reception field should be large enough for

the long text instance. When the reception field is not large enough, failure cases on long text instances happen [28, 29]. However, long text instances are widely present in various scenarios, especially in educational scenarios. Thus, some top-down methods [13, 28, 30–32] try to use the reception-field expansion methods such as [30, 31] from object detection and semantic/instance segmentation to build long-range dependency of text instances. Thus, top-down approaches usually have multiple pipelines and complex network design [33].

The bottom-up methods are advantageous for processing long or arbitrary-shape texts since bottom-up methods model text segments globally and have a more flexible representation of text contour. Moreover, the way humans read text instances also follows a bottom-up manner. The text segments in bottom-up modelling also have advantages of reflecting the character characteristics. Thus, bottom-up methods can be more robust and flexible in terms of dealing with long and arbitrary shape texts. Nevertheless, the performance of the state-of-the-art bottom-up methods [1, 28, 34] on benchmark datasets is lower than the top-down methods [2, 13, 14].

There are issues that prevent bottom-up methods from achieving their great potential. First, the bottom-up methods are more likely to accumulate errors compared with top-down approaches. Since bottom-up methods usually divide text detection into two steps, *i.e.*, the text segments proposal and the linkage reasoning. The text segments proposal step is usually a CNN based network, while the linkage reasoning step has more options such as RNNs (Recurrent Neural Networks), predefined rules, geometric location-based methods, and GCNs (Graph Convolutional Neural Networks), etc. [1, 11, 17, 18, 20, 28, 34]. Therefore, a simple connection between the text proposal step and the linkage reasoning step without rectification of false detection tends to result in error accumulation.

The error accumulation problem mentioned before may also lead to the failure of some linkage reasoning. The missing text segments will lead to unexpected gaps on a text instance, while the other text segments are of fixed shape, so it is difficult for GCNs to reason the linkage. Moreover, GCNs also require large computation resources for building the topological structure of all text segments. Considering

these, GCNs may not always be the best solution for linkage reasoning.

Moreover, some bottom-up methods [1, 11, 28, 34, 35] need to locate or sample the points on the contour and determine the correct order in which these points are visited. Namely, they need to know the connection order of the points on each text segment. However, when there are a large number of text segments, this problem is analogous to the Traveling Salesman problem, which is an NP-complete problem. Failure to solve this problem may end up in route-finding failure. Currently, some approaches try to use approximating solutions [1] or larger text segments to reduce the complexity of this problem [28, 34]. However, approximating solutions can sometimes fall into the local optimal situations, while large text segments sometimes weaken the character level information and also compromise the flexibility and the accuracy of the resultant text contours.

Thus, in Chapter 3 and Chapter 4 of this thesis, we tackle the above challenges of bottom-up arbitrary-shape scene text detection methods for achieving their great potential.

1.2.2 Navigation Tools for Educational Resources

As a practical application of arbitrary-shape scene text detection methods, the navigation tools for educational resources have an ideal expectation. Namely, the learners only need to 'click' on the knowledge points that they would like to learn in the slides, and the navigation system will automatically 'retrieve' the explanation of this knowledge point from a lengthy video. But the existing systems [8, 36–41] are unable to build a fine-grained navigation system due to a lack of the hierarchical structure information of the knowledge points. This is because even the state-of-the-art arbitrary-shape scene text detection methods [1,2,14,28,34] can only indicate the locations of text lines but not their hierarchical structure relationships. Nevertheless, a knowledge point may spread over several lines. Thus, only locating the text lines is not enough for extracting a complete knowledge point. They need to be merged according to their hierarchical structure relationship to make the knowledge points complete. Most of the existing navigation systems do not have the capability to

analyze the layout of the knowledge points, resulting in that they can only build coarse linkage between unsorted contents in the slides and explanations. Besides, the existing systems [8,36–41] do not provide particular deep-learning based solutions to detect visual entities (text, graph and formula) in the educational resources, resulting in their systems are not robust and prone to accumulated errors for dealing with the visual entities in more complex environments.

Therefore, in Chapter 5 the objective of this thesis is to propose a novel navigation tool for educational resources that can provide accurate and fine-grained navigation.

1.3 Contributions

This thesis proposes two arbitrary-shape scene text detectors addressing the challenging problems that prevent bottom-up arbitrary-shape scene text detection methods from achieving their great potential. The proposed text detectors further prove that the bottom-up methods are not inferior to, but can be even better than, the top-down methods in arbitrary-shape text detection. As the application of proposed arbitrary-shape scene text detectors, we propose a novel navigation tool for educational resources that can provide fine-grained navigation based on the restored knowledge points. The key contributions of this thesis are summarized as follows.

The first contribution is that we first excavate the relational feature of GCNs to rectify the false text segments by globally considering the "characterness" and "streamline" of text segments in the same relational structure through a weakly supervised training process in order to alleviate the false detection accumulation problem in GCN-based bottom-up methods. To the best of our knowledge, this is the first time the classification ability, instead of link prediction, of GCN is used for scene text detection. Then, we propose a novel visual-relational reasoning approach to revitalize the superb strength of typical bottom-up text-detection approaches. This is demonstrated to be effective in capturing both visual and continuity properties of the characters that determine text areas. We develop a simple but effective contour inference strategy together with the redesigned dense overlapping text segments to

depict the "characterness" of text to replace the error-prone route-founding step. This module can handle complex situations with a large number of text segments and build fine-grained topological representation for relational reasoning and text segment type classification.

The second contribution is that towards accurate detection of arbitrary-shape texts, we propose a novel bottom-up text detection method by effectively embedding deep morphology for regularizing text segments. The proposed DMOP (Deep Morphological Opening) module is embedded as a false detection removing module to reduce error accumulation. DMCL (Deep Morphological Closing) module is proposed to allow text instances of various shapes to stretch their morphology in the most significant direction while deriving their connections. The proposed deep morphological modules are trainable and they can avoid most of error-prone post-processing steps in bottom-up methods. To the best of our knowledge, this is the first time that deep morphology is introduced into the text detection area. Our method outperforms the state-of-the-art arbitrary-shape text detection methods on several highly competitive benchmark datasets.

The third contribution is that we first develop an accurate and robust deep learning based approach for extracting visual entities in educational videos. Then, we propose a visual saliency and clustering based approach to extract the hierarchical relationship among visual entities, which ensures the completeness and preciseness of visual entities. We finally propose an effective approach to align visual entities with speech texts by evaluating their semantic similarity. Here, the fine-grained matching between visual entities and their audio explanations is established. Using this fine-grained matching, we present a convenient navigation tool for educational videos that can provide the multi-level table of contents and auto-generated notes. The evaluation experiments demonstrate the effectiveness of our proposed solutions for visual entity extraction, hierarchical relationship extraction, as well as corresponding speech text matching. The user study also shows a promising improvement in the auto-generated table of contents and notes for facilitating learning.

1.4 Thesis Organization

The remaining parts of this thesis are organised as follows.

- *Chapter 2*: We review and discuss the development of the state-of-the-art bottom-up and top-down methods for scene text detection. From practical perspective, we also review and discuss the existing state-of-the-art navigation tools for educational resources.

- *Chapter 3*: We first excavate the relational feature of GCNs to rectify the false text segments by globally considering the "characterness" and "streamline" of text segments in the same relational structure through a weakly supervised training process. Then, we propose a novel visual-relational reasoning approach to revitalise the superb strength of typical bottom-up text-detection approaches. To replace the error-prone route-founding step, we further develop a simple but effective contour inference strategy together with the redesigned dense overlapping text segments to depict the "characterness" of text.

- *Chapter 4*: We develop a novel bottom-up scene text detection method using deep morphology. To reduce the false detection accumulation in the bottom-up modelling, we first propose DMOP (Deep Morphological Opening) module. This module can be embedded in both bottom-up and top-down methods as a false detection removing module. Then, to address the linkage problem, DMCL (Deep Morphological Closing) module is proposed to allow text instances of various shapes to stretch their morphology in the most significant direction while deriving their connections.

- *Chapter 5*: We first develop an accurate and robust deep learning based approach for visual entity extraction from educational videos according to their unique features. To build fine-grained matching between a visual entity and its audio explanation, we then propose a clustering approach to extract the hierarchical relationship between visual entities and use this information to associate visual entities with their corresponding audio speech text by evaluating their semantic relationship. Finally, we propose a slide-based educational

video navigation tool that is able to extract the hierarchical structure and semantic relationship of visual entities in videos by integrating multi-channel information.

- *Chapter 6*: We conclude this thesis and provide a discussion of further research.

Chapter 2

Literature Review on Scene Text Detection and Its Application in Navigating Educational Resources

In this chapter, we first present an overview of scene text detection methods in the era of deep learning. A theoretical investigation is conducted from the groundbreaking deep learning based scene text detection methods to the state-of-the-art scene text detection methods. In the overview section, we first discuss the development, design idea, and pathways of these methods. Then, we review the existing tools and systems for navigating educational resources using the detected text information as cues.

2.1 Deep Learning Based Scene Text Detection

Before the prosperity of deep learning technologies, scene text detection methods were mostly based on connected region analysis with hand-designed rules [42–44], and contained multiple independent pre-processing and post-processing steps that were easy to propagate errors. The deep learning based methods tend to establish end-to-end models and have better feature capture capability, which greatly reduces error propagation and improves the performance of scene text detection.

After the appearance of the groundbreaking deep learning based scene text detection method CTPN [17] in 2016s, the mainstream of scene text detection has become deep learning based as there is a great enhancement in the detection performance. In the beginning years of the blooming of deep learning based scene text detection, the research communities focused on dealing with multi-oriented text instances. Namely, the assumption was that the text instance followed a line structure and can be surrounded by quadrilateral bounding boxes.

However, the multi-oriented text instances are not the only form of text instances in complex natural images. There are other forms of text instances in the natural scene, namely, curved texts, which are largely seen in the advertising boards, clothing and unstable shot photos. The quadrilateral bounding boxes cannot accurately depict curved text instances and may include redundant background information. A detailed example can be found in Figure 1.2. In recent years, the research focus has changed to detecting curved text instances. Subsequently, text detection has transitioned from the detection of quadrilateral bounding boxes to the detection of contours of arbitrary shapes. TextSnake [11] is the groundbreaking work to attempt this transition.

Although the annotation methods of multi-oriented text instances and arbitrary-shaped text are different, the designing idea of multi-oriented text detectors and arbitrary-shaped text detectors generally follows two pathways, *i.e.*, top-down modelling and bottom-up modelling. The top-down modelling usually considers the text areas globally and produces the final text areas in one go. The bottom-up modelling usually considers text areas as a connected-component linkage problem, where the text segments are further combined to generate the final text instances. Both types of modelling can achieve promising results in detecting multi-oriented and arbitrary-shaped text instances and have their own advantages and disadvantages. As shown in Table 2.1, we review the different modelling types for the classic methods in detecting multi-oriented and arbitrary-shaped text instances.

2.1.1 Different Modeling of Multi-oriented Scene Text Detection

Multi-oriented Top-down Methods

The multi-oriented text instances are labelled with quadrilateral, which can be determined by four points. Thus, several methods consider the multi-oriented text detection problem as a four-point regression problem. During this time, the regression-based object detection methods such as Faster RCNN [24], SSD [25] and Yolo [26] yielded brilliant results in detecting general objects. Multi-oriented text detectors follow the structure of these regression-based object detectors. Liao *et al.*

Table 2.1 : The typical top-down and bottom-up methods designed for multi-oriented and arbitrary-shape texts.

Purpose	Modeling	Method	Venue	Backbone	Architecture
Multi-oriented Text	Top-down	TextBoxes [45]	AAAI'17	VGG16	SSD [25]
		TextBoxes++ [46]	TIP'18	VGG16	SSD
		RRPN [47]	TMM'18	VGG16	Faster RCNN [24]
		R2CNN [48]	arXiv'17	VGG16	Faster RCNN
		DDR [49]	CVPR'17	VGG16	Faster RCNN
		Synthetic [50]	CVPR'16	VGG16	Yolo [26]
		EAST [51]	CVPR'17	PVA [52]	FPN [53]
	Bottom-up	CTPN [17]	ECCV'16	VGG16	Faster RCNN
		SegLink [20]	CVPR'17	VGG16	SSD
		Pixellink [54]	AAAI'18	VGG16	FCN [27]
		ALC [55]	ACCV'16	VGG16	CNN
		NSCCE [56]	ICDAR'17	VGG16	Faster RCNN
Arbitrary-shape Text	Top-down	Mask textspotter [57]	ECCV'18	ResNet50	Mask RCNN
		ATRR [12]	CVPR'19	VGG16	Mask RCNN
		CSE [58]	CVPR'19	ResNet34	Mask RCNN
		LOMO [29]	CVPR'19	ResNet50	FPN [53]
		DB [32]	AAAI'20	ResNet50	FPN
		SDASSC [59]	TMM'21	ResNet50	CNN
		MS-CAFA [60]	TMM'21	ResNet50	CNN
		ContourNet [14]	CVPR'20	ResNet50	Mask RCNN
		Textpreceptron [61]	AAAI'20	ResNet50	FPN
		SDM [15]	ECCV'20	ResNet50	Mask RCNN
		TextFuseNet [2]	IJCAI'20	ResNet50	Mask RCNN
		PCR [62]	CVPR'21	DLA [63]	CNN
		FCENet [13]	CVPR'21	ResNet50	FPN
	Bottom-up	TextSnake [11]	ECCV'18	VGG16	UNet [64]
		PSENet [65]	CVPR'19	ResNet50	FPN
		TextRay [21]	MM'20	ResNet50	FPN
		OPOM [66]	TMM'20	ResNet50	FPN
		CRAFT [18]	CVPR'19	VGG16	UNet
		TextDragon [35]	ICCV'19	VGG16	UNet
		PuzzleNet [34]	arXiv'20	ResNet50	FPN+GCN
		DRRG [1]	CVPR'20	VGG16	FPN+GCN
		ReLaText [28]	PR'21	ResNet50	FPN+GCN

re-designed the bounding boxes regression methods in SSD and proposed two effective multi-oriented text detectors *i.e.* TextBoxes [45] and TextBoxes++ [46]. Ma *et al.* [47] proposed a multi-oriented text detector by modifying the region proposal networks in Faster RCNN. Jiang *et al.* [48] also varied the ROI-pooling size of Faster RCNN to regress four points of the multi-oriented texts. He *et al.* [49] proposed a direct regression method by addressing the drawbacks of the indirect regression in Faster RCNN and SSD. The method proposed by Gupta*et al.* [50] was inspired by the Yolo and used the random forest classifier to suppress false positive detection. EAST [51] generated four-point regression results by measuring the features from each point in the text score map. All regression-based approaches should be considered as top-down modelling methods, as the purpose of these methods is to predict a quadrilateral that covers the entire text area at once. However, regression-based methods suffer difficulty dealing with long multi-oriented texts because the anchor box used for regression may not always fit the aspect ratio of long multi-oriented texts. For example, the authors of [45] pointed out that TextBoxes might fail for text with large inter-space.

Multi-oriented Bottom-up Methods

Typical bottom-up methods divide the text into segments and then reason the link between them. Following this design structure, CTPN [17] is a pioneer work that brings text detection into the deep learning area, but it simply merges text segments according to a certain threshold and can only deal with horizontal texts. SegLink [20] is designed to detect multi-oriented texts, which aims to connect the centers of two segments with an eight-neighbourhood link prediction. Pixellink [54] treated the pixels in the text region as the text segments and grouped them according to the linkage between pixels. He *et al.* [55] proposed a text detector that aggregated local surrounding information of text segments for linking the final text area. Wang *et al.* [56] adopted a segmentation-based approach to extract the character candidates and then grouped them into text instances. These bottom-up multi-oriented text detection methods try to progressively link the text segments and are more robust for detecting longer multi-oriented texts. However, compared with the top-down

methods, the bottom-up methods usually require additional post-processing steps and are prone to accumulate errors.

2.1.2 Different Modeling of Arbitrary-shape Scene Text Detection

Arbitrary-shape Top-down Methods

The top-down methods typically consider arbitrary-shape scene text as an instance segmentation problem. Therefore, the classic framework Mask-RCNN [16] and its variations have been widely used, playing an important role in top-down methods, *e.g*, ContourNet [14], Xiao *et al.* [15], Text-RPN in Wang *et al.* [12], TextFuseNet [2], Mask textspotter [57], and Liu *et al.* [58]. For example, ContourNet [14] adopted the Mask-RCNN structure with an adaptive region proposal and constructed text contours by considering two orthogonal text maps. Xiao *et al.* [15] also modified their model based on Mask-RCNN but with a sequential deformation module before generating the mask of texts. The Text-RPN and refinement network in Wang *et al.* [12] were also related to Mask-RCNN to some extent. TextFuseNet [2] modified the feature fusion branch in Mask-RCNN, and fused word-level and weakly supervised char-level features to perform instance segmentation of text. There are also some segmentation based methods [29, 32, 61], which can be considered as top-down methods, since similar to Mask-RCNN based methods, these methods also generate mask-like text segmentation maps as their final results. In the methods shown in [29, 32, 61], the text segmentation map was constructed from a text center line and its offset to the boundary. This step is similar to the direct regression of text boundary and does not require further post-processing steps. Thus, top-down methods generally obtain text detection results from the segmentation maps.

Arbitrary-shape Bottom-up Methods

The bottom-up methods often need to connect text segments or generate text areas progressively from their centers to the border areas. These methods include TextSnake [11], CRAFT [18], Scribble Lines [67], etc. For example, TextSnake [11] decomposed text areas into circles and the text areas were generated by building connections in between as well as founding the external contour points. Following

a similar design logic, CRAFT [18] used affinity boxes to link characters, where the training was done in a weakly-supervised manner, which required char-level annotation for building affinity boxes for characters. Similar to CRAFT, Zhang *et al.* [67] also adopted the weakly-supervised approach for training the bounding boxes of characters and reasoning the linkage from a text center line. Recently, GCNs have been introduced to ensure the linkage between text segments. PuzzleNet [34], RelaText [28] and DRRG [1] are the three similar works that have utilized GCNs to link text segments. They share the same idea, *i.e.*, first using a text segment proposal network to generate text segments and then connecting text segments using GCNs. However, their methods suffer from the error accumulation problem. That is to say, if a text-like object is proposed by the text segment proposal network, the GCNs will indiscriminately link this text-like object with other text segments, resulting in error accumulation. Furthermore, the bottom-up methods [1, 11, 28, 34, 67] often require additional post-processing to determine the visiting order of the contour points. Different from the top-down methods, they cannot directly generate text regions from the segmentation maps. Although others such as [65, 68] generate text areas from the inside out with a dilation process, they also require rule-based post-processing to refine the final text contour.

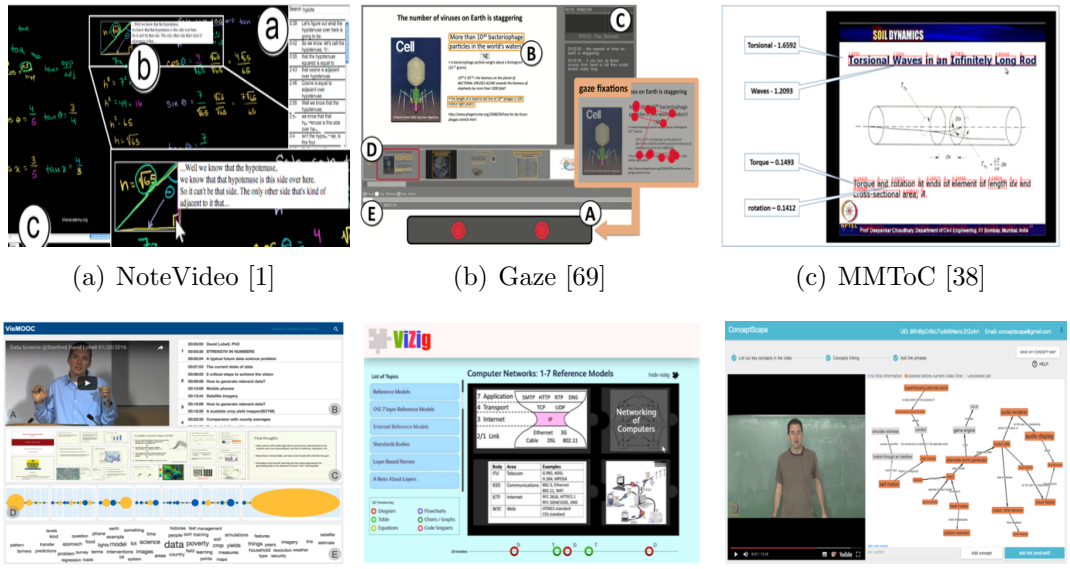
2.1.3 Summary of Modeling Methods of Scene Text Detection

The top-down methods of scene text detection consider generating a text area in one go by regressing the contour points or measuring the segmentation mask of text instances. They are more in line with the end-to-end designing idea, so they contain fewer intermediate errors. However, the design of anchors and the reception fields of the networks may limit the performance of top-down methods in detecting texts with a larger width-height aspect ratio. To expand the reception fields, top-down methods often require complex feature aggregation modules in their networks.

The bottom-up methods of scene text detection consider generating text from local pieces progressively, which are more in line with how humans read texts. The bottom-up methods are flexible and robust to represent text instances with various shapes. Compared with the top-down methods, bottom-up methods require

no anchor design and reception field expansion. However, bottom-up methods often require additional pre-processing or post-processing steps, since text segment generation and linkage reasoning steps are easier to accumulate errors. Currently, the performance of the state-of-the-art bottom-up methods [1, 28, 34] on benchmark datasets is lower than the top-down methods [2, 13, 14]. Thus, preventing error accumulation and more accurate linkage reasoning are the key steps to improving the performance of bottom-up methods.

2.2 Existing Systems and Tools of Navigating Educational Resources



(a) NoteVideo [1] (b) Gaze [69] (c) MMToC [38]

(d) System of Zhao *et al.* [8] (e) ViZig [70] (f) ConceptScape [40]

Figure 2.1 : The existing navigation systems of educational video

In the previous section, we discussed the development of the mainstream scene text detection methods. The localized text instances can be used to build navigation systems for educational videos to apply text detection methods. For slide-based educational videos, one of their characteristics is that they usually contain a large number of embedded text instances in the video channel, and rich voice information in the audio channel. The localized text instances on the video channel act as a primary cue for the navigation and indexing of educational resources. However, it

is a time-consuming task to browse through an educational video from beginning to end. Thus, navigation tools and systems that can help users locate information efficiently have received a lot of attention. As shown in Figure 2.1, system in [36] uses the user interaction data collected in the log to quickly locate the video. This system uses subtitle information to construct visual video summaries, but user information is not easy to obtain when it comes to privacy. NoteVideo [71] is a system that uses handcraft rules to quickly locate the first occurrence of the knowledge points in the video. Balasubramanian et al. [41] developed an educational video retrieval system based on video and audio channel information, which effectively summarised the content of educational videos. Nguyen *et al.* [69] developed a gaze-based system to assist note-taking on the potentially useful lecture contents (text instances) in slides. In the MMToC system, [38], the multi-channel content in educational videos was extracted to automatically create a list of topics on the table of the content of the course video. ViZig [70] built anchor points that were used to indicate specific slides on the timeline of the educational videos in their navigation system. However, ViZig assumed that each slide could only contain information in one of the six types, *i.e.*, charts, codes, diagrams, flow charts, equations and tables. Zhao *et al.* [8] presented a visual navigation system using multimodal cues that could find the lecture's explanation related to each slide. Most of the systems including [37, 39] allow users to retrieve knowledge points by a search-able text index using fused data from multiple channels. ConceptScape [40] provided a collaborative concept map that linked the concept in the speech text to a specific frame (slide) in the video according to time alignment.

However, the above-mentioned educational video navigation systems cannot perfectly correspond to each knowledge point with its explanation. This is because the existing systems [8, 36–41] do not design specific text detectors according to their task and also ignore the visual entities other than text instances. Most existing navigation systems only adopt open-sourced text detection software such as Google Tesseract* or directly use text detectors that are pre-trained from natural images,

*https://github.com/tesseract-ocr/tesseract

resulting in unstable text detection results. The text detection step in educational videos affects the quality and completeness of knowledge in navigation systems.

Chapter 3

Arbitrary-shape Scene Text Detection via Visual-Relational Reasoning and Contour Approximation

One trend in the latest bottom-up approaches for arbitrary-shape scene text detection is to determine the links between text segments using Graph Convolutional Networks (GCNs). However, the performance of these bottom-up methods is still inferior to that of state-of-the-art top-down methods even with the help of GCNs. We argue that a cause of this is that bottom-up methods fail to make proper use of visual-relational features, which results in accumulated false detection, as well as the error-prone route-finding used for grouping text segments. In this chapter, we improve classic bottom-up text detection frameworks by fusing the visual-relational features of text with two effective false positive/negative suppression (FPNS) mechanisms and developing a new shape-approximation strategy. First, dense overlapping text segments depicting the "characterness" and "streamline" properties of text are constructed and used in weakly supervised node classification to filter the falsely detected text segments. Then, relational features and visual features of text segments are fused with a novel Location-Aware Transfer (LAT) module and Fuse Decoding (FD) module to jointly rectify the detected text segments. Finally, a novel multiple-text-map-aware contour-approximation strategy is developed based on the rectified text segments, instead of the error-prone route-finding process, to generate the final contour of the detected text. Experiments conducted on five benchmark datasets demonstrate that our method outperforms the state-of-the-art performance when embedded in a classic text detection framework, which revitalizes the strengths of bottom-up methods.

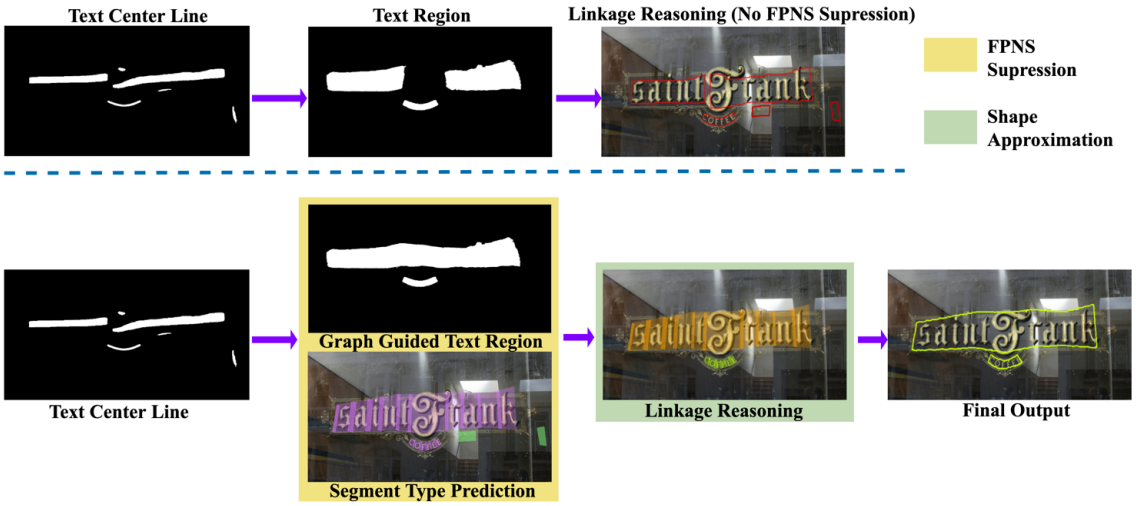


Figure 3.1 : The error accumulation problem of the existing bottom-up approaches (top) and our solution (bottom). Our Graph Guided Text Region fuses relational features with visual features and rectifies false detections. The segment type prediction module further rectifies this through excavating the "characterness" and connectivity of text segments. The Final Output shows that the false detections have been suppressed.

3.1 Introduction

Deep learning-based arbitrary-shape scene text detection methods generally follow one of two different pathways, *i.e.*, top-down approaches and bottom-up approaches. Some top-down approaches consider a text instance as a whole and estimate its text area by leveraging the segmentation results, whereas the typical bottom-up methods tend to typically divide a text instance into text segments and then group them based on certain criteria. Both types of approaches [1, 2, 14, 18, 21, 28, 34] use a feature extraction network [72, 73] with a Feature Pyramid Network (FPN) [53] as the basic text feature extraction network, and are deemed as the classic text detection framework. The major difference is that these top-down approaches try to generate more accurate segmentation masks for text areas. Consequently, more sophisticated models for object detection and semantic segmentation have been adopted to enhance the performance of text segmentation. However, these methods may not be robust for long text instances due to their limited receptive field and the insufficient geometric feature representation of CNNs [28, 29].

The bottom-up methods [1, 18, 28, 34] typically decompose a text instance into several components (*i.e.*, text segments) and then group the eligible ones together with certain purpose-designed rules. Intuitively, these methods align better with how humans read text. Additionally, the bottom-up methods often require smaller receptive fields and are therefore more flexible for modelling arbitrary-shape texts. Therefore, they may be expected to yield higher accuracy and better robustness for long and curved texts. However, the reality is often the opposite, since bottom-up methods are also prone to accumulating intermediate errors (false positives/negatives). For example, in the challenging curved text datasets CTW1500 [10] and Total-Text [9], the overall accuracy of the best-performing bottom-up methods is lower than those of best high-performing top-down methods, such as those in [2, 13, 14, 62].

To determine the constraints of these bottom-up methods, we need to understand the evolution of bottom-up methods in the deep learning era. During this period, the relationship between text segments has become more diverse and complicated due to the focus of text detection research extending from horizontal text to quadrilateral text, and then to more challenging curved text. Existing bottom-up methods (*e.g.*, [11, 17–21]) with CNN, RNN and some pre-defined heuristic rules have considered visual similarity features, sequential features and geometric features in order to connect text segments.

However, the relationship between the text segments of arbitrary-shape text is non-Euclidean, so there may still be connections between non-adjacent text segments especially in the case of curved texts. The Graph Convolutional Network (GCN) is an effective framework for representing this kind of non-Euclidean data, as it learns the representation of each node by aggregating the features from adjacent nodes [74]. In terms of capturing the relationship between text segments, state-of-the-art bottom-up methods such as DRRG [1], ReLaText [28] and PuzzleNet [34] have adopted GCNs [74]. These methods share a similar framework, namely, a text segment proposal network (*e.g.*, VGG16/ResNet50+FPN) followed by a relational reasoning (link prediction) network (*e.g.*, GCN). Nevertheless, the performance of these bottom-up methods is still lower than that of some high-performing top-down

methods [2, 13, 60].

We argue that there are two problems that need to be addressed. First, the simple connection between the text proposal network and the GCN tends to result in error accumulation. An example is shown in Figure 3.1, where the wrongly separated text (false negatives) and two text-like objects (false positives) from the FPN layer of the text segment proposal network have been further propagated after the GCN based linkage prediction. Here, the Text Region/Text Center Line results generated from the same FPN layer share similar visual features and they are often used as a double guarantee for accurate text area and text segment generation for further relational reasoning. However, sharing similar visual features sometimes leads to error accumulation. In this example, a text instance appears to be separated in both the text region map and its center line map as the FPN layer fails to build long-range dependency between the text segments in the middle. Although the relational information predicted by GCN can be treated as long-range dependency for different text segments, the simple connection of FPN and GCN does not take advantage of the relational features. Even if the text region and its centre line are both considered for generating text segments, some of the text segment candidates can still be wrongly discarded, which significantly affects the accuracy of the GCN-based linkage prediction. The errors accumulated from text regions and text centre line maps explain why the final detection result (the top-right figure) contains text instances that are incorrectly broken apart. Hence, the reasoning of text regions needs stronger relational cues to ensure their connectivity and integrity.

Moreover, the simple connections between FPN and GCN and the weight sharing design in the FPN layer result in GCNs indiscriminately treating those false positive text segments the same as other text segments in the subsequent linkage reasoning steps, leading to falsely detected text-like objects. Among the GCN based bottom-up approaches, only ReLaText [28] has attempted to randomly generate some non-text nodes so as to remove the link with the non-text nodes, but it is insufficient to suppress false positives/negatives, nor does it make advantageous use of the relational information.

Since the spaces between texts or characters are also non-text areas, both GCN and the text segment proposal network may be unable to distinguish text gaps from the non-text background. As shown from the separated text instance in Figure 3.1 (better viewed in darker colors), the separated area is right in the middle of the interval area of the text. As pointed out by [75], lacking content information increases the possibility of false detection, indicating that too large [34] or too sparse text segments [1, 28] cannot reflect the "characterness" of the content information of texts.

Another problem occurs in the text segments' final grouping stage, *i.e.*, the inference stage [1, 11, 17, 18], which also has a major impact on the final results but has not received much attention. Some bottom-up methods (*e.g.*, [1, 11, 28, 34, 35]) need to locate or sample the points on the contour and determine the correct order in which these points are visited. This is because when using the contour-drawing function such as *cv2.drawContours*, the visiting order of contour points is critical for depicting a close contour and avoiding the edge crossing problem (a route-finding failure). It should be noted that the directional search towards the ends of text instances in TextSnake [11], the sorting of the text segments' bounding boxes in TextDragon [35], the searching of the polynomial curve that fits two long sides of text instances in ReLaText [28], and the segmentation of the bisection lines based on the vertex extraction in PuzzleNet [34] can all be considered as different forms of searching for the visiting order of the contour points. This is analogous to the Traveling Salesman problem, which is an NP-complete problem. TextSnake adopted a complex heuristic rule-based route-finding method to alleviate this NP-complete problem. However, this post-processing may still end up with route-finding failure and hence is only applicable to specific conditions. The DRRG [1] used a greedy Min-Path algorithm to give an approximate solution to this problem. However, approximating solutions can sometimes fall into local optimal situations (see Figure 3.2) when there are too many contour points. PuzzleNet and ReLaText adopted larger text segments to control the number of contour points so as to reduce the complexity of this NP-complete problem. However, larger text segments sometimes

(a) Route-finding failure from [1] (b) Our shape-approximation solution

Figure 3.2 : Route-finding gives suboptimal visiting order when there are too many contour points.

weaken the character level information of text and also compromise the flexibility and accuracy of the resultant text contours. How to find text contours without diminishing the "characterness" of texts and introducing suboptimal failures needs to be further explored.

In this chapter, we propose designing the text segments in a dense and partially overlapping manner so as to retain the "streamline" characteristic of text, which allows flexible connection as much as possible to adapt text instances of arbitrary shapes. To ensure that the text segments are able to depict the "characterness" of text, text segments are designed to be fine enough to reflect characters and the spacings between them (see the segment type definition in Figure 3.4). Such a dense, overlapping design of text segments also benefits the construction of the subsequent fine-grained character-to-character graph structure, which enables character-level relational representation for further relational reasoning.

Moreover, to address the simple connection and error accumulation issues, we re-examine GCN's other capabilities to further classify the types of text segments. The existing methods [1, 18, 28] have used the linkage reasoning ability of GCN but ignored its node classification ability. When text segments are represented as nodes of a graph, segments of the same text instance are those who share similar "characterness" and "streamline" properties. The feature aggregation step in graph reasoning can also differentiate true positive text segments by relational reasoning. This establishes a backward feedback mechanism that can retrieve and suppress false

detections (including both false positives and false negatives) during link prediction. Classifying the types of segments enables the GCN to examine the content information for more accurate text region prediction. This forms one of our two false positive/negative suppression (FPNS) strategies.

Secondly, we develop another FPNS strategy that aggregates the text's visual and relational features to suppress false detection, as shown in Figure 3.1. Instead of considering only visual text feature maps from the output of FPN layers, here we propose a Location-Aware Transfer (LAT) module to convert relational features produced by GCNs into visual compatible features. Then, a Fusion Decoding (FD) module is introduced to fuse the relational features with the visual features to generate a Graph Guided Text Region (GGTR) map. Since relational features are a ready-made high-dimensional representation of long-range dependency based on the relational connection of text segments, the GGTR map provides additional long-range dependency to guide the connectivity and integrity of proposed text regions and suppress false detection, resulting in more potential true positive text segments. The GGTR map forms the other FPNS strategy to ensure enough candidate text segments are generated for further graph reasoning. Thus, the two FPNS strategies decompose the overall text map rectification problem into the problem of rectifying text segments to improve their fault tolerance.

Finally, instead of the error-prune route-finding process, we develop a shape approximation (SAp) strategy to group the rectified text segments and approximate the contour of the text. Such a design allows an unlimited number of text segments to approximate the contours of arbitrary-shape text.

In summary, the main contributions of our work are as follows.

1) We utilize the relational feature of GCNs to rectify text segments by globally considering their "characterness" and "streamline" in the same relational structure through a weakly supervised training process. To the best of our knowledge, this is the first time the classification ability, instead of link prediction, of GCNs is used for scene text detection.

2) We propose a novel visual-relational reasoning approach to increase the feature discriminability for falsely detected text segments in typical bottom-up arbitrary-shape text-detection approaches and take advantage of their strengths. This is demonstrated to be effective in capturing both visual and continuity properties of the characters that determine text areas.

3) We redesign the text segments in a dense and partially overlapping manner and develop a simple but effective contour inference strategy to depict the "characterness" of text, which can handle complex situations of arbitrary-shape text with enhanced relational reasoning and type classification capabilities.

Experiments conducted on several curved text detection benchmark datasets show that our approach surpasses the state-of-the-art performance and demonstrates the strength of bottom-up approaches for arbitrary-shape scene text detection. It is also shown that bottom-up methods are not necessarily inferior to, but can be even better than, top-down methods in arbitrary-shaped text detection.

3.2 Related Work

3.2.1 GCNs based Bottom-up Arbitrary-shape Text Detection

To further excavate the relationship between text segments, researchers used GCN to integrate geometric features, visual features, and relational features for link prediction [1, 28, 34]. These three methods share a similar framework but differ in the size and number of text segments. PuzzleNet [34] used the largest text segments to merge adjacent text segments with angle differences less than $\pi/36$, which was inflexible in the following segments grouping stage. RelaText [28] generated smaller text segments compared with PuzzleNet, but these segments were still too large and too sparse to depict the character features of the text. The size of the text segments in DRRG [1] is the smallest. Although the segments contain more char-level features and are more flexible when grouping, the complexity of their route-finding process increases significantly. Thus, there is often a trade-off between the size and number of text segments and a difficulty of the grouping operation, so that they need to be further explored.

In our approach, we use 'characterness' and 'streamline' to accurately depict the character-level and adjacency-level features of text content, to avoid the computational overload of the route-finding process with a shape approximation (SAp) strategy while greatly increasing flexibility.

3.2.2 False Detection Suppression

Suppressing false detection requires distinguishing text and non-text areas accurately. The first way is to strengthen the network's capability for depicting text features by adding additional text content information. Some methods try to use char-level annotation to increase the 'characterness' expression. For example, CRAFT [18] adopted a weakly-supervised training strategy to train the character region score to split words into characters, and TextFuseNet [2] fused word-level, char-level and global-level features of text to conduct text instance segmentation using Mask-RCNN [16]. Similarly, Zhang *et al.* [67] also used this framework to obtain char-level annotation with a weakly-supervised training strategy. However, training char-level annotation as an instance segmentation task means doing detection and recognition together, and hence brings an additional computational burden. Other methods such as [13, 15, 28, 32, 34, 59, 66, 67] try to insert blocks like Non-local [30] or Deformable Convolution [31] to increase the network's capability for extracting text features, but these tricks often bring limited improvement if we look at their detection results. We need to focus more on the characteristic of the text itself.

The other way is to jointly consider multiple text maps to get a final text area. ContourNet [14] tried to suppress non-text areas by considering text maps in horizontal and vertical directions, which effectively suppressed false positives. Nevertheless, their method still suffers from false negatives when both horizontal and vertical text maps have defects. TextSnake [11] and DRRG [1] predicted an additional Text Center Line (TCL) map and multiplied it with the original Text Region (TR) map for False Positives/Negatives Suppression (FPNS). As shown in Figure 3.1, both TCL and TR are resulted from FPN. It is very likely that both TCL and TR maps are flawed, which may result in error accumulation. Moreover, simply setting a threshold on different text maps [14] or directly multiplying different text maps increases

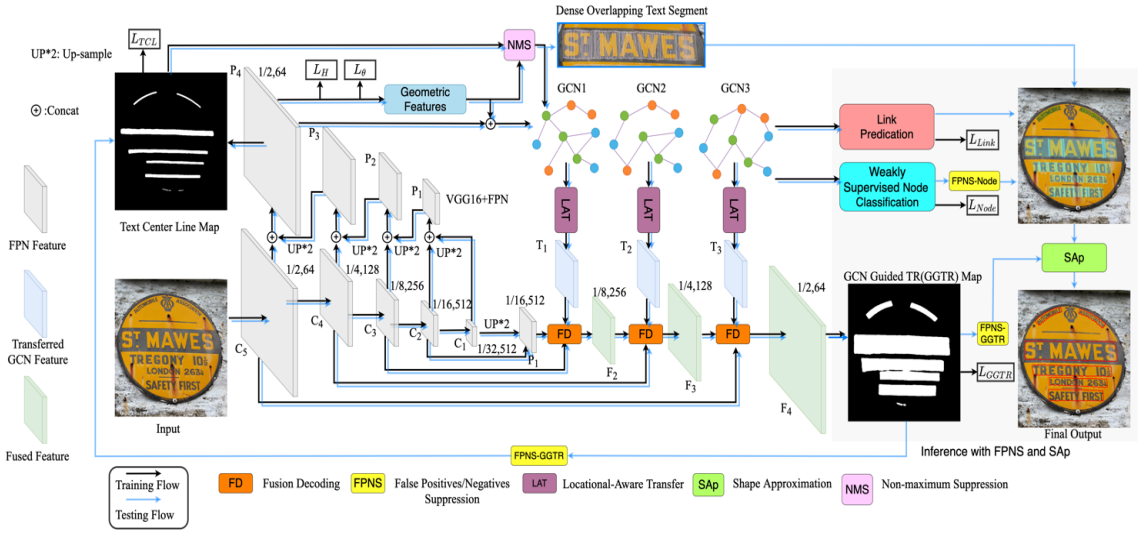


Figure 3.3 : The overall structure of the proposed network.

the risk of having the true positive text areas suppressed as well. It requires to have a more effective and accurate approach to each text instance on text maps and indirectly use both of them.

3.3 Methodology

The overall structure of our network is illustrated in Figure 3.3. We adopt VGG16 plus FPN for visual feature extraction. Visual features C_5 to C_1 are first extracted and up-sampled to P_1 to P_4. P_4 is used to generate TCL maps and geometric features of possible dense overlapping text segments. P_4 and the geometric features are embedded into nodes and then converted into a graph structure. Then, the text segment types are annotated by a weakly supervised network. The details of the text segment generation, annotation steps and weakly supervised training are provided in Sect. 3.3.1. Sect. 3.3.2 describes the deployment of three GCN layers to learn the graph representation between text segments, where the linkage relationship and the text segment types are reasoned by aggregating the features of adjacent nodes in the graph structure.

Meanwhile, in Sect. 3.3.3, the visual feature P_1 is enhanced by aggregating with the relational feature produced by GCNs to provide additional long-range depen-

dency. Since visual feature and the relational features have different dimensions, we develop firstly a Location-Aware Feature Transfer module to align the two types of features and secondly a Fusion Decoding (FD) module to fuse them. Finally, we obtain a Graph Guided Text Region (GGTR) map based on the visual-relational features.

During the training stage, the training of the TCL map, GGTR map, geometric features, link prediction and node classification is guided by the loss function designed in Sect. 3.3.4. During the inference stage (see Sect. 3.3.5 for details), the GGTR map that contains visual-relational features is used to rectify the TCL map and contour generating process for suppressing false positive/negative text segments. The weakly supervised node classification results are used to further rectify the false positive/negative text segments. Finally, a novel text-instance-based contour inference module is used to approximate the contour of the rectified text segments to obtain the final results.

3.3.1 Dense Overlapping Text Segments

Text segments are basic components of many bottom-up methods and have therefore been used in existing works. In this chapter, our re-designed dense overlapping text segments play an important role in our method, since they are involved in the generation of graph structure that is used in GCNs and the proposed shape approximation strategy. In the following sub-sections, we first give the details of the text segments. Then, the ground-truth type of each text segment is obtained in a weakly supervised manner.

Text Segment Generation

We first adopt VGG16+FPN for generating the TCL map, as well as the geometric features for each text segment. The text segments are defined as small rectangles by (x, y, h, w, θ), where x and y represent the X and Y coordinates of the rectangle's center point, and h, w and θ represent the height, width and rotation angle of the rectangle, respectively.

As shown in Figure 3.3, the final extracted features P_4 are used to generate the TCL map, and the geometric features (*i.e.*, Height map H and Angle map Θ) are used to restore the geometric representation of text segments. More specifically, $Conv_{1\times1}$ is applied on P_4 to obtain these text maps, namely, TCL, H and Θ. The Width map (W) is obtained by applying a clip function to the Height map while keeping the width w between 2-6 pixels, as shown in Figure 3.4.

Note that, the width w of text segments affects the generation of the graph structure and the final contour shape, and hence influences the overall performance of text detection. This is because our constructed visual representation for each text segment is designed to depict the "characterness" and "streamline" of the text instance. Since characters in a text instance are usually taller than they are wide, denser text segments with a smaller width can therefore approximate the contour of text instances more accurately using their connected shapes. Also, smaller text segments better differentiate characters, character spacing and word spacing in the text, which will be used to further suppress false detections. The ablation studies conducted in Sect. 3.4.4 demonstrate this with extended discussion on the impact of the width w on the final results.

Moreover, the ground truth of the TCL map is obtained by shrinking the ground truth text area vertically from top and bottom towards the middle line using the method described in [1, 11]. To generate dense overlapping text segments, here we first generate small rectangles centered at each text pixel of the TCL map, with their height, width and angle based on the Height, Width and Angle maps. Then, a non-maximum suppression (NMS) algorithm with an IoU threshold of 0.5 is applied on this TCL map to remove those heavily overlapping rectangles. This step balances the computational load and ensures the density and connectivity of the text segments, so it increases the flexibility for them to fit the contours of text of various shapes.

Weakly Supervised Text Segment Annotation

After we have obtained the dense overlapping text segments, the next step is to determine their types. The type information of text segments enables them to

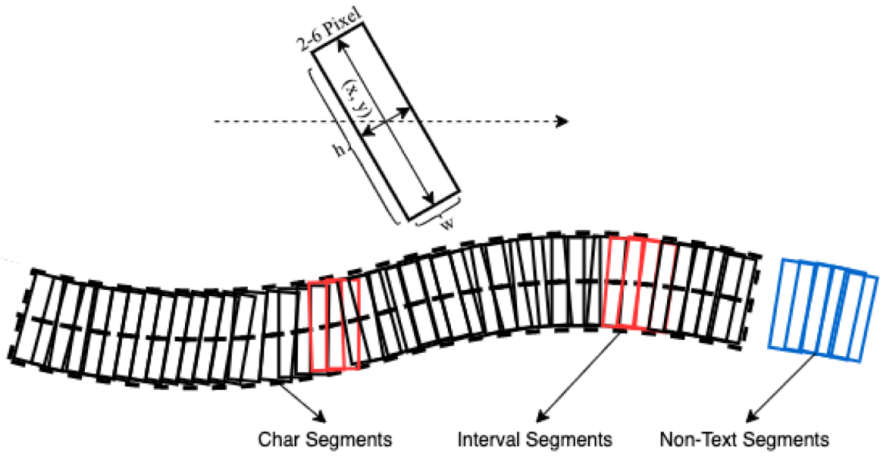


Figure 3.4 : The three types of text segments.

better reflect text's "characterness" and allows building a character-to-character graph structure for further fine-grained relational reasoning and rectification. In our work, text segments are categorized into one of three types: Char Segments, Interval Segments and Non-Text Segments (see Figure 3.4). Char Segments contain characters or partial characters only, Non-Text Segments are background, and Interval Segments are the gaps or spacing between characters or words.

Interval Segments can often be confused with Non-Text Segments, especially in text instances with large inter-character spaces, so they need to be classified under close supervision. However, none of the real-world datasets provide annotation for the intervals between characters. Therefore, in our approach, we utilize a synthetic dataset "SynthText" [50] that has both character-level and word-level annotations, and propose a weakly-supervised training approach to classify the type of text segments.

To start the weakly supervised learning, labelled samples are first generated from the SynthText dataset. Specifically, let us denote the i-th text segment with the center point coordinates (x_i, y_i) as S^i, the collection of all text segments as S^*, the collection of Char Segments as S^*_{char}, the non-character segment set $S^* - S^*_{char}$ as $S^*_{\neg char}$, the collection of all Interval Segments as S^*_{inter}, and the union of all character areas in the image as C. With these initial samples, we can train a preliminary

node classifier, which is then fine-tuned on real text datasets with only word-level annotation to learn the types of text segments in an iterative, weakly-supervised manner.

During the iterations of the weakly-supervised learning, the node classifier may misclassify some Char Segments as Interval Segments. Subsequently updating the loss based on these false Interval Segments will confuse the training and seriously affect the performance of the learned model. Therefore, we consider text segments' adjacent segments and use a simple approach to ensure only those more reliable Interval Segments back-propagate the loss.

Towards this, we first define the distance between each Char Segment in S^*_{char} and non-char segment in $S^*_{\neg char}$ as the Euclidean distance of their centers:

$$d(S^i_{char}, S^j_{\neg char}) = \|(x_i, y_i) - (x_j, y_j)\|_2 \,, \tag{3.1}$$

where $\forall S^i_{char} \in S^*_{char}$ and $\forall S^j_{\neg char} \in S^*_{\neg char}$.

To alleviate the situation of updating the loss based on these false Interval Segments, for each Char Segment S^i_{char}, we only perform gradient update for its closest Interval Segment S^i_{inter}, which is the non-char segment that has the minimal Euclidean distance from S^i_{char}, *i.e.*,

$$S^i_{inter} = \left\{ S^k_{\neg char} \middle| d(S^i_{char}, S^k_{\neg char}) = \min_{j \le |S^*_{\neg char}|} d(S^i_{char}, S^j_{\neg char}) \right\}. \tag{3.2}$$

It is worth noting that, here although only the closest non-char segments are defined, as the Interval Segments go through the gradient update process, those non-char segments further away from the Char Segments may still have a high chance of being correctly classified as Interval Segments after the weakly supervised training. The Interval Segments shown in Figures 3.1 and 3.5 demonstrate this.

Furthermore, since there is no char-level ground-truth or internal ground-truth available in the existing real text datasets, in our approach, we use a simple mechanism (or conditions) to determine whether a weakly supervised training result is

acceptable. The conditions are that the union of the predicted Char Segments and Interval Segments covers at least 90% of the annotated text areas and the annotated text areas do not contain Non-Text Segments. Thus, with the iteration of training, gradually more and more text segments are correctly classified until most classified Char Segments and Interval Segments are in annotated text areas.

With the weak supervision it is likely that some Interval Segments are misclassified as Char Segments (and vice versa), especially when the characterness properties of the characters are rather weak (*e.g.*, blurry, tiny, etc). However, our main concern is the correctness of distinguishing Non-Text Segments and Interval Segments to prevent the text instance being mistakenly broken apart due to the misclassification of some Interval Segments. The exemplar image in Figure 3.3 (see the area between "Y" and "10" in the image) shows an example of a large interval space, which, if misclassified may result in the text instance being broken apart. Therefore, as long as they are not misclassified as Non-Text Segments (*i.e.*, background), such misclassification will not cause errors, as our FPNS strategies only suppress those segments classified as Non-Text Segments.

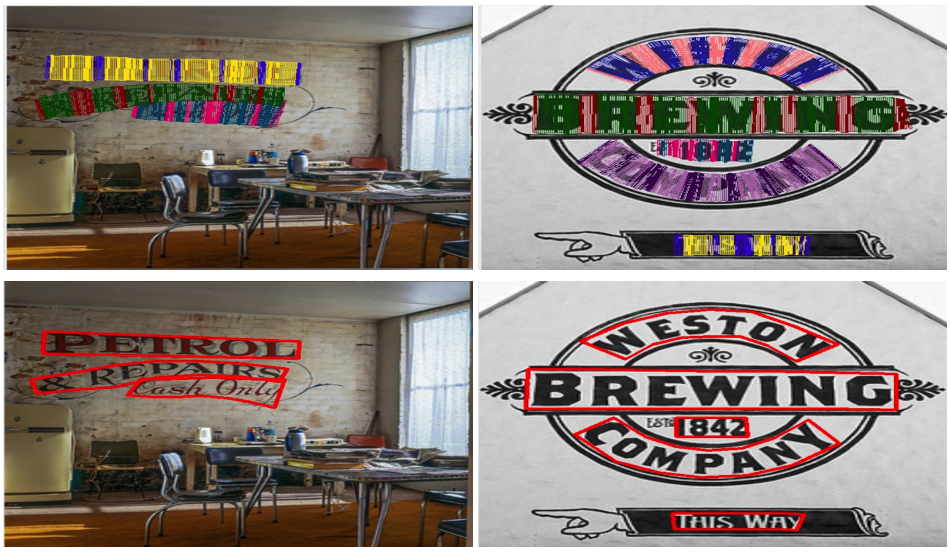


Figure 3.5 : The results of weakly supervised annotation (1st row) and the ground truth (2nd row).

3.3.2 Graph based Reasoning

We now give details about building the relational representation between text segments, as well as link prediction and node classification, after the dense overlapping text segments have been obtained.

To predict the links between text segments and classify their types, we first need to integrate the feature representation of each text segment and construct their graph structures.

We consider two types of features for text segments, *i.e.*, the visual feature F_{vis} is obtained by applying convolutional operation on P_4 in FPN for generating the TCL map and the geometric feature F_{geo}. We note F_{seg} as the feature representation matrix of text segments. Following the methods in [1, 34], the RRoI Align [47] is adopted to pool the visual features from FPN. F_{geo} is formed by concatenating the feature maps TCL, H, W and Θ, and is embedded into the same space as F_{vis} as:

$$F_{seg} = [F_{vis}(P_4); F_{geo}([TCL; H; W; \Theta])], \tag{3.3}$$

where the $[;]$ operator indicates concatenation.

To build the graph structure of text segments, we treat each text segment as the pivot and connect its eight closest text segments (measured by the Euclidean distance of their center points) as its 1-hop nodes. Then, for each 1-hop node, we connect its four closest text segments (excluding the pivot).

The pivot, its 1-hop nodes and 2-hop nodes compose the basic graph structure for the text segment, which can be represented by an adjacency matrix $\mathcal{A}$. We consider the top three closest text segments of a pivot to have a link with the pivot. Similar to those in [1, 74, 76], the graph convolution is represented by:

$$F_{seg}^{(i+1)} = \sigma([F_{seg}^{(i)}; \tilde{\mathcal{D}}^{\frac{1}{2}}\tilde{\mathcal{A}}\tilde{\mathcal{D}}^{\frac{1}{2}}F_{seg}^{(i)}]W^{(i)}), \tag{3.4}$$

where the $F_{seg}^{(i+1)}$ is the feature representation for the $(i+1)$-th layer after the graph convolution of $F_{seg}^{(i)}$, $\tilde{\mathcal{D}}^{\frac{1}{2}}\tilde{\mathcal{A}}\tilde{\mathcal{D}}^{\frac{1}{2}}$ is the re-normalization trick from the original GCN

paper [74] and can be calculated by $\mathcal{A}$, σ is a non-linear activation function and $W^{(i)}$ is the weight of the graph convolutional layer.

Thus, the relational link prediction can be written as:

$$\hat{r} = Softmax(m^{(i+1)}(F_{seg}^{(i+1)})), \tag{3.5}$$

where $m^{(i+1)}$ denotes a multi-layer perceptron with a PReLU activation. The feature matrix F_{seg} can also be written as $F_{seg} = [f_1, f_2, ...f_v, ...]^T$, where f_v is the feature representation vector for the single text segment v.

The GCN segment classification results for the text segment v can be represented by:

$$\hat{f}_v = Softmax(\varphi(\frac{1}{|\mathcal{N}(v)|}\sum_{u\in\mathcal{N}(v)}\mathcal{W}f_u + b)), \forall v \in \mathcal{V}, \tag{3.6}$$

where $\mathcal{N}(v)$ means the neighbor segments in the graph structure $\mathcal{V}$ of text segment v, $\mathcal{W}$ and b are the weights of the graph convolution, and φ is the non-linear activation function. Eq. (3.6) is the node-level graph feature aggregation of Eq. (3.4).

After feature aggregation, the relational reasoning results can be used to combine text segments that have a strong relationship. The weakly supervised node classification results are used to refine the detection results inherited from the previous FPN layer. Thus, the text segment type reasoning prevents detection errors from being further accumulated. Both link prediction and node classification make inferences based on the high-dimensional relational features on the same local graph, and can benefit each other when sharing the same weights. The node classification steps further classify Text Segments and Non-Text Segments, reducing the number of false text segments being incorrectly linked in link prediction.

3.3.3 Visual-Relational Feature Fusion

The relational features from GCN layers provide a ready-made high-dimensional representation of long-range dependency depicting the relational connection of text segments. We fuse them with the visual features to complement the long-range

dependency between text segments for generating graph-guided text regions.

However, the dimensionality of the relational features is different from that of the visual features, so these two types of features cannot be fused directly. To address this issue, we propose Graph features' Location-Aware Transfer (LAT) to reconstruct the graph convolutional feature of i-th GCN layer F^i_{seg} in Eq. (3.4) with consideration of the location information of each node in the graph. The relational features after GCN contain the structural information of the graph in its first two dimensions. For example, the relational feature $F^i_{seg} \in \mathbb{R}^{N\times M\times D}$, where N is the total number of nodes (*i.e.*, text segments) in the input image, M is the number of 1-hop nodes, and D is the dimension of GCN features and it is similar to that of the channel space in the convolutional layer. Our goal is to reshape the F^i_{seg} of dimension $N \times M \times D$ to $C \times H \times W$, where C, W and H are the channel, width and height of the input image. The core idea is to fill in the graph features in the corresponding region of the transferred map based on the location information of each text segment, i.e., building connections of the nodes on the graph structure with their location on the image, as the location of each node is also translation invariant compared to the visual features. The transferred features $T_i \in \mathbb{R}^{C\times H\times W}$ for F^i_{seg} after LAT can be calculated with Algorithm 1.

Algorithm 1: Location-Aware Transfer (LAT)

Input: F^i_{seg}, F_{geo}
Output: Transferred feature T_i
$T_i = ZeroLike(H, W, C)$
$N = F^i_{seg}.shape[0]$
for $k \leq N$ **do**
  $(x_k, y_k, h_k, w_k, \theta_k) = Geometric(F_{geo}, S^k)$
  if $type(S^k) \neq nontext$ **then**
    for $\forall x, y \in bbox(x_k, y_k, h_k, w_k, \theta_k)$ **do**
      $T_i[x, y, :] = F^i_{seg}[k, :, :].flatten()$
    end
    $T_i = T_i.transpose(2, 0, 1)$
  end
end
return T_i

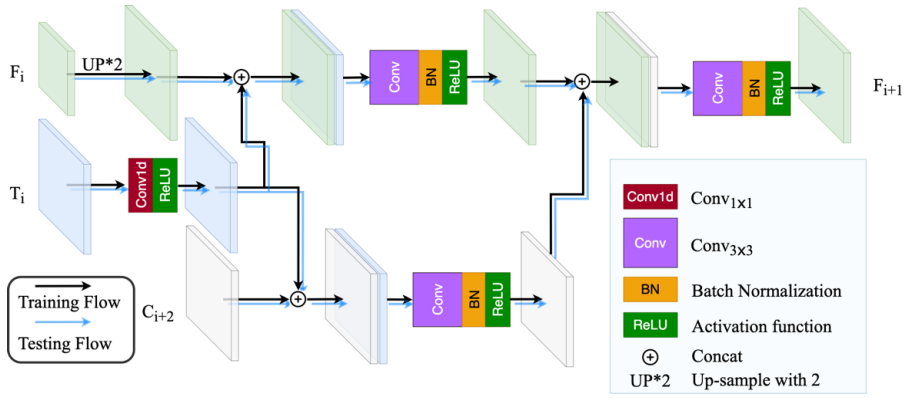


Figure 3.6 : The network structure of multi-modal fusion decoding module.

First, we calculate the geometric feature $(x_k, y_k, h_k, w_k, \theta_k)$ of text segment S^k by the geometric feature map F_{geo}. Here, we only consider transferring the graph features of char/Interval segments. Then, we can find the bounding box (bbox) of each text segment on the feature map of T_i according to $(x_k, y_k, h_k, w_k, \theta_k)$. For each point (x, y) in the bbox, we fill the flattened graph feature into T_i.

Since FPN is unable to reason the relationship between different text segments, we introduce a multi-modal Fusion Decoding (FD) module to capture the long-range dependency between individual text regions according to their relational features as well as FPN features. The final goal is to generate a Graph Guided Text Region (GGTR) map for further FPNS.

As shown in Figure 3.6, the proposed FD module produces the final fused feature F_{i+1} as:

$$F' = \begin{cases} CBR[UP(P_i); CR(T_i)] & \text{if} \quad i = 1 \\ CBR[UP(F_i); CR(T_i)] \end{cases} \tag{3.7}$$

$$F_{i+1} = CBR[F'; CBR[(C_{i+2}); CR(T_i)]], \tag{3.8}$$

where CBR denotes $Conv_{3\times3}$, Batch Normalization and ReLU, CR donates $Conv_{1\times1}$ and ReLU, and UP denotes two times of up-sampling. Finally, we apply the $Conv_{1\times1}$ on F_3 to obtain a GGTR map, which will be used as a visual-relational guide in the inference stage.

3.3.4 Objective Function

The overall objective function consists of six parts and is formulated as:

$$L = L_{GGTR} + L_{TCL} + L_H + L_\theta + L_{Link} + L_{Node}, \tag{3.9}$$

where L_{GGTR} and L_{TCL} are the cross-entropy losses from the GGTR and TCL maps, respectively, and L_H and L_θ are Smoothed $L1$ losses for Height and Angle. In our work, OHEM [77] is adopted for training L_{TCL}, and the ratio of positive sample and negative sample areas is set to $1:3$. The linkage loss L_{Link} is also cross-entropy between the ground truth of the linkage relationship between text segments and $\hat{r}$ in Eq. (5.5).

For the text segment classification loss L_{Node}, since our annotation is obtained with weakly supervised training, we only consider the text segments that are labeled as Char Segments, Interval Segments and Non-Text Segments (denoted as f_c^v, f_i^v, and f_n^v respectively) and ignore those that have not been labeled. Thus,

$$L_{Node} = -(\sum_{v \in \mathcal{N}} f_c^v \log \hat{f}_c^v + \sum_{v \in \mathcal{M}} f_i^v \log \hat{f}_i^v + \sum_{v \in \mathcal{O}} f_n^v \log \hat{f}_n^v), \tag{3.10}$$

where $\mathcal{N}$, $\mathcal{M}$, and $\mathcal{O}$ are the collections of all Char Segments, Interval Segments and Non-Text Segments, respectively.

3.3.5 Inference by FPNS and Shape-Approximation (SAp)

The Graph Guided Text Region map and the text segment classification results from the GCN are then used to rectify the TCL map for removing false detection. The relationship prediction and the dense overlapping text segments together ensure the completeness and accuracy of using the contour of the grouped text segments to approximate the contour of text. The connection of the text areas is ensured by the 'characterness' and 'streamline' of our dense overlapping text segments. The overlapping of text segments also reflects their connectivity, ensuring that the shape of the grouped segments can be used to approximate the contour of the original text instances accurately. In our work, instead of jointly rectifying multiple text maps

Algorithm 2: Inference with FPNS and shape-approximation

Input: Image, GGTR map, TCL map, H map, W map, Θ map, GCN_{node} and GCN_{link}
Output: Text Contours $\mathcal{C}$
$\mathcal{C} = List()$
$TCL' = TCL \bigcap Shrink(GGTR)$ ▷ FPNS (GGTR)
$TCLContours = cv2.findContours(TCL')$
$\mathcal{S} = List()$
for $contour \in TCLContours$ **do**
  $Segments = Getboxes(TCL', H, W, \Theta, contour)$
  $Segments = NMS(Segments, 0.5)$
  for $seg \in Segments$ **do**
    $type = GCN_{node}(seg)$
    if $type == nontext$ **then**
      $Segments.remove(seg)$ ▷ FPNS (Node)
    end
    $\mathcal{S}.append(Segments)$
  end
end
$\hat{\mathcal{S}} = GCN_{link}(\mathcal{S})$
$Blankimage = ZeroLike(Image)$
for $ConnectedSegments \in \hat{\mathcal{S}}$ **do**
  for $box \in ConnectedSegments$ **do**
    $cv2.drawContours(Blankimage, box)$
  end
  $Blankimage = (Blankimage \oplus B_{3\times3}) \ominus B_{3\times3}$
  $Contour = cv2.findContours(Blankimage)$
  if $IoU(Blankimage, Blankimage * GGTR) >= 0.5$ **then**
    $\mathcal{C}.append(Contour)$ ▷ FPNS (GGTR)
  end
end
return $\mathcal{C}$

as in [1,11,14], we decompose the false positive/negative suppression problem based on the grouped text segments reasoned by visual-relational feature. Therefore, our approach does not use element-wise multiplication or search the entire map, so it avoids accidentally removing true positive text areas.

Algorithm 2 illustrates our proposed shape-approximation algorithm. The inputs require the text maps of GGTR, TCL, H, W and Θ, as well as node classification GCN_{node} and link prediction components GCN_{link} from the GCN. First, we perform a union operation on the TCL map and shrink the GGTR map to produce a rectified TCL map TCL'. The text area of the GGTR map is shrunk from the top and bottom towards the middle until single-pixel height is reached. With this step, the GGTR map is introduced to provide the long-range dependency of text

segments and also provide more candidates for the graph reasoning stage. The function $cv2.findContours$ is used to find the potential text centre lines of the text instances. Following that, for each text instance, the *Getboxes* function is used to generate dense text segments for pixels in the rectified TCL map. The classifications from GCN_{node} are used here to suppress false detection of text segments, and the remaining text segments are stored in the list $\mathcal{S}$. Then, the link prediction components from GCN_{link} are used to restore the order of text segments that are grouped into $ConnectedSegments$ based on their linkage relationships.

For shape approximation (SAp), we first draw the text segments on a blank image and then use the contour of these overlapping text segments to approximate the original text contour. To connect text segments with each other more closely, an opening operation with a kernel size of 3×3 is performed. Similar to [1, 11], the GGTR map is used to further suppress the text instances with IoUs less than 0.5.

It is worth noting that our inference method discards the route-finding process of existing bottom-up methods for determining the visiting order of text contour points. Instead, we generate dense text segments to approximate the text contour more accurately with only one NMS processing step.

3.4 Experiments

To evaluate the performance of our proposed method, comprehensive experiments are conducted on five mainstream scene text detection datasets, including the curved text datasets Total-Text [9] and CTW1500 [10], the multi-oriented dataset ICDAR2015 [78] and MSRA-TD500 [79], and the multi-lingual dataset ICDAR2017-MLT [80]. For a fair comparison, VGG16 and ResNet50 are used as the feature extraction backbone to exclude the performance gain brought by the backbone. Moreover, only single-scale testing results are compared to exclude the performance gain contributed by multi-scale testing.

In this section, we first introduce the datasets tested in the experiments and the implementation details. Then, quantitative and qualitative comparisons with the state-of-the-art approaches and ablation studies are presented to show the superb

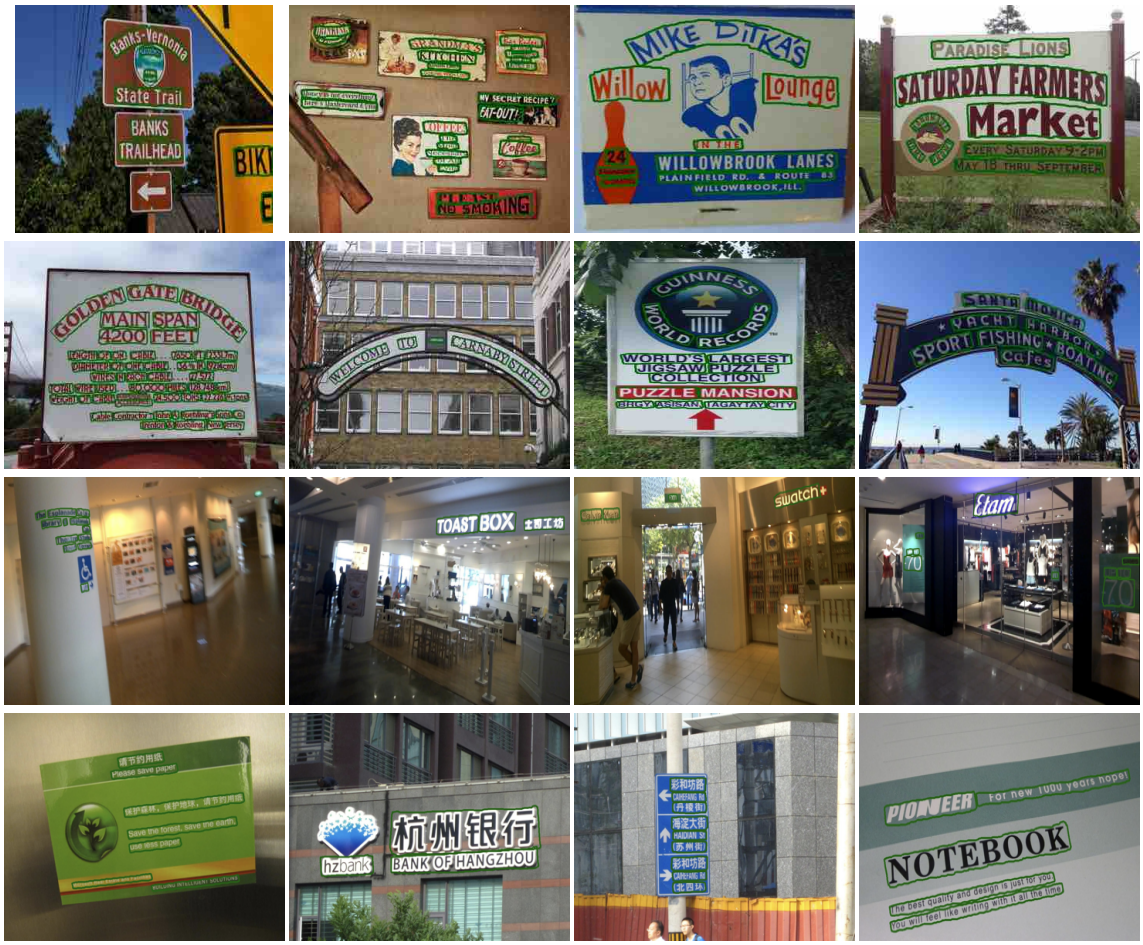


Figure 3.7 : Visualization of the text detection results obtained on CTW1500 (1st row), Total-Text (2nd row), ICDAR2015 (3rd row) and MSRA-TD500 (4th row).

performance of our method and the effectiveness of each of our proposed approaches.

3.4.1 Datasets

SynthText [50] consists of 800,000 synthetic images of multi-oriented text with char-level annotation. The char-level annotation is used to produce the ground truth of text segment types.

Total-Text [9] consists of horizontal, multi-oriented, and curved text instances with polygon and word-level annotations. It contains 1,255 training images and 300 testing images.

CTW1500 [10] consists of curved and multi-oriented texts, all annotated by polygons, and tends to label long curved text. It has 1,500 training and 1,000 testing

images.

ICDAR2015 [78] includes multi-orientated and small-scale text instances. Its ground truth is annotated with word-level quadrangles. It contains 1,000 training and 500 testing images.

MSRA-TD500 [79] is dedicated to detecting multi-oriented long non-Latin texts. It contains 300 training images and 200 testing images with word-level annotation. Here, we follow the previous methods [28, 32] and add 400 training images from TR400 [81] to extend this dataset.

ICDAR2017-MLT [80] is designed for detecting multi-lingual texts of nine languages with word-level annotation. It contains 9,000 training images and 9,000 testing images.

3.4.2 Implementation Details

Our algorithm is implemented using PyTorch 1.7. The VGG16 is pre-trained with ImageNet, with FPN adopted for multi-scale feature extraction. We conduct our experiments on NVIDIA RTX 3090 GPU with 24GB memory. All images used for training and testing are of a single scale. For training, the images are resized to 640×640 for images from CTW1500 and Total-Text, and 1280×1280 for images from ICDAR2015, MSRA-TD500 and ICDAR2017-MLT. Data augmentation techniques including rotation, random cropping, color variations, adding random noise, blur, and changing lightness are adopted. The batch size is set to 10.

We first use SynthText to pre-train our model for 10 epochs using Adam optimizer and a learning rate of 0.001. Then, we fine-tune our model on real text benchmark datasets for 800 epochs with SGD optimizer and a learning rate of 0.001. The momentum of SGD is set to 0.9. For testing, the short side of images is kept at 640 pixels for CTW1500 and Total-Text, and 1,280 pixels for ICDAR2015, MSRA-TD500 and ICDAR2017-MLT, while retaining their aspect ratios.

3.4.3 Comparison with State-of-the-art Methods

Examples of the text detection results obtained with our proposed method are visualized in Figure 3.7. In this section, we present the quantitative comparisons of the

detection results with the state-of-the-art approaches on all mainstream benchmark datasets, *i.e.*, CTW1500, Total-Text, ICDAR2015, MSRA-TD500 and ICDAR2017-MLT, respectively. The "-" in the table indicates that the comparative method did not report their results on the dataset.

Curved text detection: Our results on CTW1500 and Total-Text, the two most representative benchmark datasets of arbitrary shape text, demonstrate the superb effectiveness of our algorithm dealing with the highly curved and spatially separated texts. As shown in Table 3.1 and Table 3.2, our method achieved the F-measure scores of 86.4% and 87.6% on these two datasets, respectively, and outperformed all existing state-of-the-art methods. Specifically, our method remedies the shortcomings of the bottom-up methods [1, 28, 34] and can achieve state-of-the-art results with only a VGG16 backbone without introducing deformable convolutional blocks to increase the backbone network's feature extraction ability as in [13, 28, 32, 59, 66]. Compared with the state-of-the-art top-down methods [13, 15], our method has gained improvement of 0.9 percent on CTW1500 and 0.7 percent on Total-Text respectively over the best-performing top-down methods, thanking to our proposed false-detection suppression and shape-approximation strategies, as well as the dense overlapping design of text segments. Different from [2] requiring a weakly supervised instance segmentation to predict the mask and class of each character, our method focuses on classifying the types of text segments without extra segmentation.

The SOTA results obtained on these two datasets further support our claim that the underachieved results of the existing bottom-up methods are not caused by the limited feature capturing ability of the text proposal backbones or GCN. Also, the bottom-up methods are not inferior, but can be even superior, to the top-down methods when adopting our proposed false-positive/negative suppression and the shape-approximation strategies in the inference stage.

Multi-oriented text detection: On the dataset ICDAR2015, as shown in Table 3.3, our method also achieved a comparable SOTA result of an F-measure of 89.5%, outperforming some high-performing top-down methods such as FCENet [13]

Table 3.1 : Results on CTW1500. (P: Precision, R: Recall, F: F-measure, †: bottom-up methods, §: top-down methods, *: GCN methods)

Method	Venue	Backbone	P(%)	R(%)	F(%)
TextSnake† [11]	ECCV'18	VGG16	67.9	85.3	75.6
LOMO§ [29]	CVPR'19	ResNet50	85.7	76.5	80.8
TextRay† [21]	MM'20	ResNet50	82.8	80.4	81.6
OPOM† [66]	TMM'20	ResNet50	85.1	80.8	82.9
DB§ [32]	AAAI'20	ResNet50	86.9	80.2	83.4
CRAFT† [18]	CVPR'19	VGG16	86.0	81.1	83.5
TextDragon† [35]	ICCV'19	VGG16	84.5	82.8	83.6
SDM§ [15]	ECCV'20	ResNet50	85.8	82.2	84.0
PSE† [82]	CVPR'19	ResNet50	84.8	79.7	82.2
PAN§ [83]	ICCV'19	ResNet50	86.4	81.2	83.7
*PuzzleNet† [34]	arXiv'20	ResNet50	84.1	84.7	84.4
*DRRG† [1]	CVPR'20	VGG16	85.9	83.0	84.4
PCR § [62]	CVPR'21	DLA34 [63]	87.2	82.3	84.7
Dai et al.§ [59]	TMM'21	ResNet50	86.2	80.4	83.2
*ReLaText† [28]	PR'20	ResNet50	86.2	83.3	84.8
ContourNet§ [14]	CVPR'20	ResNet50	85.7	84.0	84.8
FCENet§ [13]	CVPR'21	ResNet50	87.6	83.4	85.5
Dai et al.§ [60]	TMM'21	ResNet50	85.7	85.1	85.4
TextFuseNet§ [2]	IJCAI'20	ResNet50	85.0	**85.8**	85.4
*Ours†	-	VGG16	**89.8**	83.3	**86.4**

and ContourNet [14] by more than 2.5 percentage points.

Our result is slightly lower than that of the TextFuseNet method in [2]. We argue that TextFuseNet adopted an instance segmentation strategy, which introduced a classification (recognition) step to enhance detection results. It is also noted that the approaches of [2, 14, 28, 34] have adopted the multi-scale training or testing strategy, which is a widely-known tip used on ICDAR2015 that can boost the detection accuracy significantly on this dataset. Whereas, we only use single-scale training and testing rather than multi-scale training or testing for fair comparison.

On the dataset MSRA-TD500, as shown in Table 3.4, our method also achieved the SOTA result with an F-measure of 87.0%. Moreover, thanking to our proposed FPNS strategy, our method achieves the highest recall rate of 83.8%. This shows the effectiveness of our idea of depicting the streamline of the texts, which helps to

Table 3.2 : Results on Total-Text. (†: bottom-up methods, §: top-down methods, *: GCN methods)

Method	Venue	Backbone	P(%)	R(%)	F(%)
TextSnake† [11]	ECCV'18	VGG16	82.7	74.5	78.4
LOMO§ [29]	CVPR'19	ResNet50	88.6	75.7	81.6
TextRay† [21]	MM'20	ResNet50	83.5	77.8	80.6
OPOM† [66]	TMM'20	ResNet50	88.5	82.9	85.6
DB§ [32]	AAAI'20	ResNet50	87.1	82.5	84.7
CRAFT† [18]	CVPR'20	VGG16	86.0	81.1	83.5
TextDragon† [35]	ICCV'19	VGG16	85.6	75.7	80.3
SDM§ [15]	ECCV'20	ResNet50	89.3	84.7	86.9
PSE† [82]	CVPR'19	ResNet50	84.0	79.0	80.1
PAN§ [83]	ICCV'19	ResNet50	89.3	81.0	85.0
*DRRG† [1]	CVPR'20	VGG16	86.5	84.9	85.7
PCR § [62]	CVPR'21	DLA34 [63]	88.5	82.0	85.2
SDASSC § [59]	TMM'21	ResNet50	85.4	81.2	83.2
*ReLaText† [28]	PR'20	ResNet50	84.8	83.1	84.0
ContourNet§ [14]	CVPR'20	ResNet50	86.9	83.9	85.4
MS-CAFA § [60]	TMM'21	ResNet50	84.6	78.6	81.5
FCENet§ [13]	CVPR'21	ResNet50	89.3	82.5	85.8
TextFuseNet§ [2]	IJCAI'20	ResNet50	87.5	83.2	85.3
*Ours†	-	VGG16	**89.4**	**85.8**	**87.6**

retrieve some miss-detected texts.

As our approach focuses on detecting arbitrary-shape text, it is a common situation that arbitrary-shape text detectors cannot significantly improve the SOTA results in both multi-oriented detection benchmarks, especially ICDAR2015, which exhibits large scale variations of texts. For example, the arbitrary-shape text detectors [1, 13, 14, 28, 34, 59, 60, 62, 66] can only achieve SOTA results on at most one multi-oriented detection benchmark. The arbitrary-shape text detectors focus more on the space variation of texts, whereas multi-oriented text detectors focus more on scale variation.

Multi-lingual text detection: As shown in Table 3.5, on the multi-lingual scene text dataset ICDAR2017-MLT, our network also surpassed the SOTA method OPOM [66] with a Precision of 82.9%, a Recall of 74.0% and an F-measure of 78.2%. This is because the dense design of the text segments can effectively depict the 'char-

Table 3.3 : Results on ICDAR2015. (†: bottom-up methods, §: top-down methods, *: GCN methods)

Method	Venue	Backbone	P(%)	R(%)	F(%)
TextSnake† [11]	ECCV'18	VGG16	84.9	80.4	82.6
LOMO§ [29]	CVPR'19	ResNet50	91.3	83.5	87.2
OPOM† [66]	TMM'20	ResNet50	89.1	85.5	87.3
DB§ [32]	AAAI'20	ResNet50	91.8	83.2	87.3
CRAFT† [18]	CVPR'19	VGG16	89.8	84.3	86.9
TextDragon† [35]	ICCV'19	VGG16	**92.5**	83.8	87.9
SDM§ [15]	ECCV'20	ResNet50	88.7	88.4	88.5
PSE† [82]	CVPR'19	ResNet50	86.9	84.5	85.7
PAN§ [83]	ICCV'19	ResNet50	84.0	81.9	82.9
*PuzzleNet† [34]	arXiv'20	ResNet50	89.1	86.9	88.0
*DRRG† [1]	CVPR'20	VGG16	88.5	84.7	86.6
SDASSC § [59]	TMM'21	ResNet50	87.2	81.3	84.1
ContourNet§ [14]	CVPR'20	ResNet50	86.1	87.6	86.9
MS-CAFA § [60]	TMM'21	ResNet50	86.2	82.7	84.4
FCENet§ [13]	CVPR'21	ResNet50	90.1	82.6	86.2
TextFuseNet§ [2]	IJCAI'20	ResNet50	91.3	**88.9**	**90.1**
*Ours†	-	VGG16	91.4	87.7	89.5

acterness' property that non-Latin texts also exhibit. Moreover, our proposed FPNS and SAp strategies enable the network to accurately identify the connectivity among the multi-lingual texts. The highest F-measure demonstrates that our method has good stability in detecting multi-lingual texts.

3.4.4 Ablation Studies

To verify the effectiveness of our proposed FPNS and SAp strategies, ablation studies are conducted on CTW1500, Total-Text, ICDAR2015 and MSRA-TD500 datasets.

Table 3.6 shows the comparison results. Here, we use 'P', 'R' and 'F' to represent Precision, Recall and F-measure, respectively. 'Node' refers to the node classification based FPNS (false positive/negative suppression) strategy and ' GGTR' refers to the GGTR based FPNS strategy. 'SAp' denotes our proposed shape-approximation strategies. The baseline is the GCN-based bottom-up method with contour route-

Table 3.4 : Results on MSRA-TD500. (†: bottom-up methods, §: top-down methods, *: GCN methods)

Method	Venue	Backbone	P(%)	R(%)	F(%)
TextSnake† [11]	ECCV'18	VGG16	83.2	73.9	78.3
OPOM† [66]	TMM'20	ResNet50	86.0	83.4	84.7
DB§ [32]	AAAI'20	ResNet50	**91.5**	79.2	84.9
CRAFT† [18]	CVPR'19	VGG16	88.2	78.2	82.9
PAN§ [83]	ICCV'19	ResNet50	84.4	83.8	84.1
*PuzzleNet† [34]	arXiv'20	ResNet50	88.2	83.5	85.8
*DRRG† [1]	CVPR'20	VGG16	88.1	82.3	85.1
PCR § [62]	CVPR'21	DLA34 [63]	90.8	83.5	**87.0**
*ReLaText† [28]	PR'20	ResNet50	90.5	83.2	86.7
*Ours†	-	VGG16	90.5	**83.8**	**87.0**

Table 3.5 : Results on ICDAR2017-MLT. (†: bottom-up methods, §: top-down methods, *: GCN methods)

Method	Venue	Backbone	P(%)	R(%)	F(%)
LOMO§ [29]	CVPR'19	ResNet50	78.8	60.6	68.5
OPOM† [66]	TMM'20	ResNet50	82.9	70.5	76.2
DB§ [32]	AAAI'20	ResNet50	83.1	67.9	74.7
CRAFT† [18]	CVPR'19	VGG16	80.6	68.2	73.9
SDM§ [15]	ECCV'20	ResNet50	**84.2**	72.8	78.1
PSE† [82]	CVPR'19	ResNet50	73.8	68.2	70.9
*DRRG† [1]	CVPR'20	VGG16	74.5	61.0	67.3
MS-CAFA § [60]	TMM'21	ResNet50	79.5	66.8	72.6
*Ours†	-	VGG16	82.9	**74.0**	**78.2**

finding [1].

A few observations can be made from Table 3.6 as follows.

i) The GCN-Node-classification-based FPNS, *i.e.*, 'FPNS (Node)', takes weakly-supervised GCN node classification results into account and brings 0.4%, 0.2%, 0.4% and 0.4% gains of F-measure on CTW1500, Total-Text, ICDAR2015 and MSRA-TD500 datasets, respectively. The GGTR based FPNS fuses multi-modal features and brings 0.5%, 0.5%, 0.8% and 0.6% gains of F-measure on the tested datasets. When both FPNS strategies (*i.e.*, Node classification based and GGTR based FPNS) are adopted, they bring 1.5%, 1.2%, 1.2% and 1.1% gains of F-measure on the tested

Table 3.6 : The impact of our proposed FPNS and SAp strategies.

Datasets	GCN	Node	GGTR	SAp	P (%)	R (%)	F (%)
CTW1500	✓	×	×	×	85.9	83.0	84.4
	✓	✓	×	×	87.6	82.3	84.8
	✓	×	✓	×	87.4	82.6	84.9
	✓	×	×	✓	87.2	83.1	85.1
	✓	✓	✓	×	88.6	**83.4**	85.9
	×	×	✓	✓	88.6	82.6	85.5
	✓	✓	✓	✓	**89.8**	83.3	**86.4**
Total-Text	✓	×	×	×	86.5	84.9	85.7
	✓	✓	×	×	86.8	85.1	85.9
	✓	×	✓	×	87.7	84.7	86.2
	✓	×	×	✓	87.6	85.0	86.3
	✓	✓	✓	×	88.9	85.0	86.9
	×	×	✓	✓	87.9	84.8	86.3
	✓	✓	✓	✓	**89.4**	**85.8**	**87.6**
ICDAR2015	✓	×	×	×	88.7	86.4	87.5
	✓	✓	×	×	89.0	86.8	87.9
	✓	×	✓	×	88.8	87.9	88.3
	✓	×	×	✓	89.0	88.3	88.6
	✓	✓	✓	×	90.5	87.0	88.7
	×	×	✓	✓	91.0	87.6	89.3
	✓	✓	✓	✓	**91.4**	**87.7**	**89.5**
MSRA TD500	✓	×	×	×	88.2	83.0	85.5
	✓	✓	×	×	88.9	83.1	85.9
	✓	×	✓	×	89.6	82.8	86.1
	✓	×	×	✓	89.0	83.6	86.2
	✓	✓	✓	×	89.8	83.6	86.6
	×	×	✓	✓	89.9	83.4	86.5
	✓	✓	✓	✓	**90.5**	**83.8**	**87.0**

datasets. We argue that the further performance gains are due to that the objectives of GGTR and node classification well align with each other in terms of effectively removing text-like interference and therefore the two operations benefit each other.

ii) Our proposed shape-approximation strategy 'SAp' has improved the F-measure by 0.7%, 0.6%, 1.1% and 0.7% on the tested datasets, because it avoids the problem of the route-finding process being trapped at a local maximum.

iii) Furthermore, when both FPNS and SAp strategies are used, together they have achieved significant improvements on all tested datasets by 2.0%, 1.9%, 2.0% and

1.5%, respectively. This is because false positive/negative suppression removes those noisy segments that affect the shape-approximation, resulting in more accurate text regions.

The results are interpreted as follows. The FPNS (GGTR) contains two parts in our work. The first part calculates the union between the GGTR map and the TCL map, resulting in more potential text segment candidates than directly multiplying the two maps containing visual features together (such as in methods [1, 11]). The second part removes false detections by measuring IoU with the GGTR map. Both rectify false detections from a visual-relational perspective, as the GGTR map results from visual-relational fusion. FPNS (Node) rectifies false detections by measuring attributes of the text segments in local graph structures and upgrades GCNs to a multiple-task network rather than only linkage reasoning, modifications which support each other. Both node classification and link prediction utilize the same relational features and boost each other's performance. This explains why the overall performance is often further improved when both FPNS (Node) and FPNS (GGTR) are applied. The performance improvements reflect that our FPNS strategies can suppress false detections while not overly affecting true detections.

Similarly, the improved recall brought by SAp can be attributed to that the proposed SAp uses the contour of the dense text segments to approximate the contour of a text instance. This avoids the typical routing-finding failure when getting stuck in local sub-optimums that causes miss-detection. Moreover, when all three strategies are combined, both of the SAp's closing operation and FPNS (Node) affect the shape and size of how text segments are grouped during the FPNS (GGTR) process. The results show that our method is effective in terms of suppressing false positives/negatives. Moreover, the node classification step in FPNS (Node) upgrades GCNs to a multiple-task network rather than only linkage reasoning. Both node classification and link prediction take the same relational feature from GCNs and boost each other's performance. This is a reason why the overall performance is often further improved when both FPNS (Node) and FPNS (GGTR) are applied.

The Impact of GCN

We also conducted additional experiments with and without using GCN, while keeping FPNS and shape-approximation. In this ablation experiment, the TR map was obtained from the original FPN layer, whose weight is shared with the TCL map, without the guidance of GCN. The results listed under the heading 'GCN' show that even without GCN, our proposed FPNS and SAp strategies can still produce F-measure improvements of 1.1%, 0.6%, 1.8% and 1.0%, respectively.

We argue that GCN makes limited contributions when dealing with text instances that are spatially close enough to each other. Such text instances are more common in multi-oriented texts, because most text segments in these texts do not have many spatial changes and have relatively small character and word spaces. Moreover, for the non-Latin multi-oriented texts, there is little or no space between characters in many texts. In this case, the dense overlapping design of the text segments and our shape-approximation strategy are sufficient to ensure connectivity of text segments. This further demonstrates that the proposed FPNS and SAp strategies are able to boost the performance of bottom-up methods with visual reasoning cues when GCN is removed. However, as shown in Table 3.6, using only visual reasoning, (FPNS (GGTR)+SAp) and relational reasoning (GCN+SAp) performs worse than using both visual-relational reasoning cues. The results show that visual-relational reasoning cues produced 0.2%-1.3% performance improvement on several benchmark datasets compared to other reasoning cues, demonstrating that the visual-relational reasoning cues can further boost the performance.

Long-range Dependency on CTW1500

Table 3.7 : The impact of the annotation types on the proposed method.

IoU	CTW1500 (long-text)			Total-Text		
	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
0.5	89.8	83.3	86.4	84.8	75.9	80.1
0.7	78.7($\downarrow_{11.1}$)	71.7($\downarrow_{11.6}$)	75.0($\downarrow_{11.4}$)	69.9($\downarrow_{14.9}$)	60.0($\downarrow_{15.9}$)	64.4($\downarrow_{15.7}$)

To determine the effect of our approach on long-range dependency, we performed

ablation studies on CTW1500, which has long text instances (see Table 3.6). Long-range dependency is achieved mainly by fusing the relational features of GCNs with visual features to generate the GGTR map. Therefore, with/without using FPNS ("GGTR" in Table 3.6) reflects the two situations of with/without the additional LAT + FD modules to capture long-range dependency. The improvements of F-measure obtained on the datasets with line level annotation (CTW1500 and TD-500) also show that fusing the visual and relational features has contributed to capturing long-range dependency.

Additionally, we compared the quality of the detection results obtained in the Total-Text and CTW1500 datasets with the same settings when the threshold of the IoU was lifted from 0.5 to 0.7. As shown in Table 3.7, applying a more strict matching criterion (*i.e.*, IoU = 0.7) reduced the detection performance on both datasets. However, the reduction in the long-text annotated CTW1500 is smaller than that in Total-Text. This shows that our method favors long-text instances.

The Impact of the Width of Text Segments

In order to assess the impact of the width of text segments on the detection accuracy, we conduct an additional ablation study on the CTW1500 dataset, as this dataset contains long curved texts that are sensitive to the setting of text segment width. Table 3.8 shows that a width between 2-6 pixels achieved the highest F-measure of 86.4%. It can also be seen that when the width of text segments is larger than 6 pixels, the F-measure decreases. We argue that the wider the text segments are, the less well they reflect the characteristics of text, *e.g.*, various spaces between

Table 3.8 : The impact of the width of the text segments on the detection accuracy obtained on CTW1500.

Width (pixels)	P(%)	R(%)	F(%)
1-3	**90.0**	82.7	86.2
2-6	89.8	**83.3**	**86.4**
8-12	89.7	83.0	86.2
16-24 [1]	88.3	82.3	85.2
$\geqslant$24 [28, 34]	87.7	82.6	85.1

text and characters, and that this affects the flexibility of the grouping process, resulting in poorer performance. Although a width of 1-3 pixels has achieved slightly higher precision of 90.0%, there are more text segments generated, increasing the burden of NMS. Therefore, in our experiments the width of text segments was set to 2-6 pixels.

The Effectiveness of LAT and FD

In this work, to realize the proposed visual-relational feature reasoning and capture the long-range dependency between text segments, relational features obtained from GCNs are fused with the visual features obtained from the FPN layers. To address the dimensional difference between the relational and visual features, in Section 3.3.3, we proposed an LAT module to reconstruct the graph convolutional features and a multi-modal Fusion Decoding (FD) module to fuse them. These two modules together generate the GGTR map to regularize text segments. To demonstrate the effectiveness of the LAT and FD, we conducted additional experiments on CTW1500. The results are shown in Table 3.9.

Note that, in FPNS (GGTR), the GGTR map is used to rectify false positive/negative text segments, as it is the final visual-relational representation of applying LAT and FD. The GGTR map is also indirectly involved in the process of FPNS (Node), as it rectifies the text segments before FPNS (Node), leading to the change of relational structure between text segments. Hence, evaluating the two FPNS mechanisms (as shown in the Table 3.6) is equivalent to the evaluation when LAT+FD are in place. FD cannot be applied and tested independently without the LAT module because the LAT process is an essential step for fusing the visual features from FPN and the relational features from GCNs. Instead, we tested the effectiveness of FD by replacing the FD module with simple concatenation. As shown in Table 3.9, the FD module can effectively fuse visual and relational features for further FPNS processing. The LAT module with simple concatenation still brings 0.8% gain (improved from 84.4% to 85.2%) in F-measure, as the relational information of GCNs is reused in capturing long-range dependency. When the FD module replaces simple concatenation, there is also a further 0.7% gain (improved from 85.2% to

Table 3.9 : The effectiveness of LAT and FD on CTW1500. (w/o: without)

Module	P(%)	R(%)	F(%)
w/o LAT	85.9	83.0	84.4
LAT + Concatenation	87.4	83.2	85.2
LAT + FD	**88.6**	**83.4**	**85.9**

Table 3.10 : The effectiveness of the proposed FPNS mechanism on suppressing false positives and false negatives.

FPNS	TPs	FPs	FNs	P (%)	R (%)	F (%)
×	2546	418	522	85.9	83.0	84.4
✓	**2559**↑	**329**↓	**509**↓	**88.6**↑	**83.4**↑	**85.9**↑

85.9%) in F-measure. Compared to simple concatenation, the proposed FD module fuses the visual and relational features more effectively.

The Quantitative and Qualitative Analysis of FPNS Strategies

To quantitatively and qualitatively validate the effectiveness of the proposed FPNS on suppressing false positives and false negatives, ablation studies are conducted on CTW1500, which contains rich samples of long, curved and multi-oriented texts. **i) Quantitative analysis.** We first examine the numbers of false positive samples (FPs), false negative samples (FNs) and true positive samples (TPs) in the detection results with and without applying our proposed FPNS strategies. As shown in Table 3.10, both FPs and FNs have been successfully suppressed to some extent after applying the proposed FPNS strategies, while TPs increase. This is also reflected by the increase of the Precision (P) and Recall (R), and consequently the F-measure (F). Table 3.6 shows that after applying the proposed FPNS strategies, both Precision and Recall have surpassed the SOTA results. **ii) Qualitative analysis.** Figure 3.1 shows an example where false positives and false negatives are suppressed with the proposed FPNS strategies. We owe this performance gain to that the proposed FPNS are supervised by the training loss of GCNs, and are less likely to affect true positive samples when suppressing false positives and false negatives.

Table 3.11 : Comparison of detection results with different backbones

Backbone	CTW1500			Total-Text			ICDAR2015		
	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
ResNet50	89.4	**83.5**	86.3	**89.8**	84.8	87.2	**92.0**	86.6	89.2
VGG16	**89.8**	83.3	**86.4**	89.4	**85.8**	**87.6**	91.4	**87.7**	**89.5**

The Impact of the Different Backbones

For assessing the impact of the backbone network to the overall performance of our model, we conduct additional studies on CTW1500, Total-Text and ICDAR2015. The results are shown in Table 3.11.

As it shows, when equipped with VGG16, our proposed method has obtained slightly better results than with ResNet50. The same phenomenon has also been reported in [84, 85].

Figure 3.8 : Some failure cases (1st row) and the ground truth (2nd row).

Time Efficiency

Finally, we also report the efficiency of the proposed method. Note that, different scales of the input images and the total number of text segments affect the running

time. We collect the statistics of the average processing time per image in each dataset on a workstation with a single NVIDIA Quadro P6000 GPU with an Intel(R) Xeon(R) Gold 5122 CPU. The NVIDIA Quadro P6000 is an NVIDIA GTX 1080Ti-equivalent GPU, and the latter is commonly used to report time efficiency in the area of scene text detection. The running time reported in Table 3.12 shows that our model can achieve decent running time. The running time in the curved text detection datasets is faster than those in multi-oriented datasets, since the texts in the latter one are typically smaller than those in the curved text benchmarks. As we limit the training to single scale, we have to increase the size of the images to ensure that tiny texts are detected, so the inference time is increased to some extent.

Table 3.12 : The time efficiency of the proposed method.

GPU	Datasets	Input Size	Inference Time	NMS Time	Total Time
Quadro P6000	CTW1500	640	0.28s	0.51s	0.79s
	Total-Text	640	0.22s	0.58s	0.80s
	ICDAR2015	1280	0.68s	0.89s	1.57s
	MSRA-TD500	1280	0.67s	0.90s	1.57s
	ICDAR2017-MLT	1280	0.61s	0.69s	1.30s

3.4.5 Limitation

Our method can handle texts with large word space thanking to the densely overlapping design of text segments as well as GCN. However, failure cases happen when long text is separated by non-text objects (see the first and second images in Figure 3.8). Currently, neither top-down nor bottom-up methods can handle this situation well as the texts are visually separated and how to present the results is debatable even for humans. A better solution is to reasoning according to the semantic information of text. Moreover, failure cases may happen on some text-like objects or super tiny texts, and these are also common challenges for other state-of-the-art methods [2, 13, 62]. Examples of such failure cases are shown in Figure 3.8.

3.5 Summary

In this chapter, we have analyzed the main reasons why the existing bottom-up text detection methods are underperforming compared with the top-down methods in arbitrary shape text detection. Aiming to eliminate the problem of error accumulation, we have proposed false positive/negative suppression strategies that take visual-relational feature maps into account to make grouping inference of densely designed text segments with regards to GCN's node classification and relational reasoning ability. A simple but effective shape-approximation module has been designed to replace the error-prone route-finding process, which is currently widely adopted in bottom-up methods. The state-of-the-art results obtained on several public datasets have demonstrated the effectiveness of our approach on arbitrary-shape text detection, which further proved that the bottom-up methods are not inferior to, but surpass the top-down methods with the benefits of false positive/negative suppression in combination with shape-approximation.

Chapter 4

MorphText: Deep Morphology Regularized Accurate Arbitrary-shape Scene Text Detection

Bottom-up text detection methods play an important role in arbitrary-shape scene text detection but there are two restrictions preventing them from achieving their great potential, *i.e.*, 1) the accumulation of false text segment detections which affects subsequent processing, and 2) the difficulty of building reliable connections between text segments. Targeting these two problems, we propose a novel approach, named "MorphText", to capture the regularity of texts by embedding deep morphology for arbitrary-shape text detection. Towards this end, two deep morphological modules are designed to regularize text segments and determine the linkage between them. First, a Deep Morphological Opening (DMOP) module is constructed to remove false text segment detections generated in the feature extraction process. Then, a Deep Morphological Closing (DMCL) module is proposed to allow text instances of various shapes to stretch their morphology along their most significant orientation while deriving their connections. Extensive experiments conducted on four challenging benchmark datasets (CTW1500, Total-Text, MSRA-TD500 and ICDAR2017-MLT) demonstrate that our proposed MorphText outperforms both top-down and bottom-up state-of-the-art arbitrary-shape scene text detection approaches.

4.1 Introduction

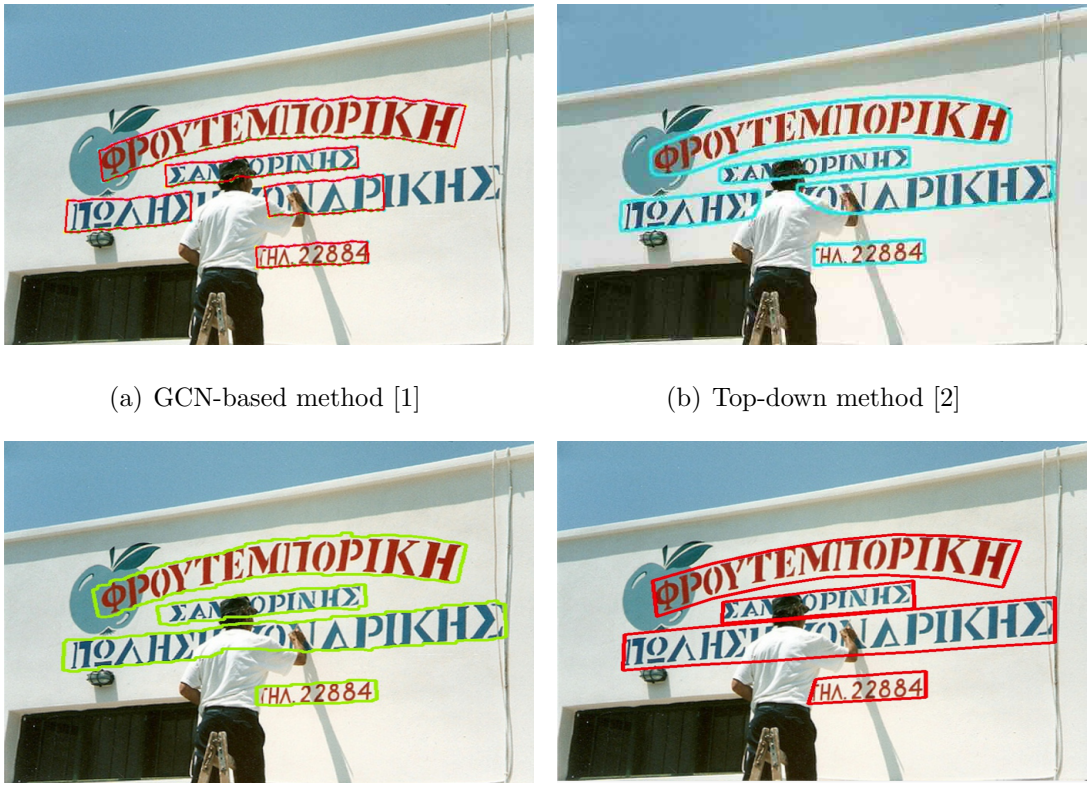
The mainstream approaches for arbitrary-shape scene text detection can be grouped into two types: top-down and bottom-up. The top-down approaches usually model the text areas globally and produce the final text areas in a single pass, while bottom-up methods usually model text areas as a connected-component link-

age problem.

The top-down methods align with the current design strategy of end-to-end deep learning models, so they often require fewer post-processing steps compared with the bottom-up methods. Moreover, some successful modules designed for object detection and semantic/instance segmentation can be directly embedded into top-down methods to enhance detection performance. However, texts differ significantly from general objects in terms of scale, aspect ratio, etc. Some direct embedding operations have not considered the characteristics of the texts themselves. As a consequence, top-down methods sometimes fail to detect long texts due to the limited reception field of CNNs [28, 29]. This has triggered some reception-field expansion methods, such as the non-local [30] and deformable convolution [31] modules in [13, 28, 32]. However, their improvement to arbitrary-shape text detection is limited.

Bottom-up methods are naturally advantageous for processing long or arbitrary-shape texts, since the rules for forming words and sentences in the real world intrinsically follow a bottom-up manner. Bottom-up methods can be more robust in terms of dealing with long texts when the connections between text segments are handled properly. Nevertheless, there are two issues that prevent them from achieving their great potential.

First, the bottom-up methods are more likely to accumulate false positive detections. This is because bottom-up methods usually need to combine feature representations generated from different modalities, *e.g.*, the visual representation of individual text segments as well as the relational representation between text segments. The processes of learning the visual representation of individual text segments and learning the relational representation between different text segments are usually independent or partially independent of each other and are therefore difficult to be optimized simultaneously. For example, some state-of-the-art bottom-up methods [1, 28, 34] have explored embedding Graph Convolutional Neural Networks (GCNs) for learning the relational information. However, the training of the visual representation and GCN modules are partially independent, so their gradient update may also be partially independent from each other. Moreover, the cross-modality



(a) GCN-based method [1]

(b) Top-down method [2]

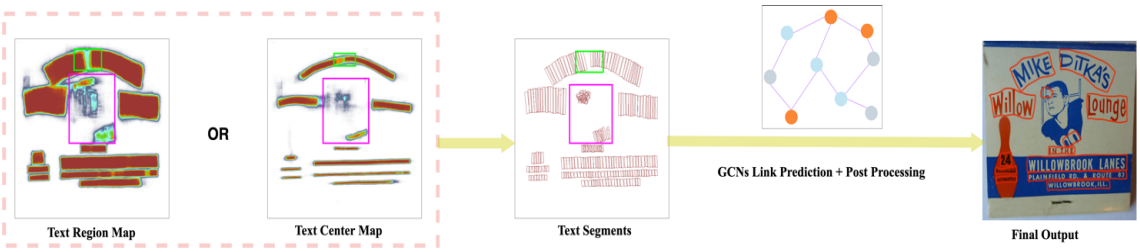
(c) Our MorphText

(d) Ground truth

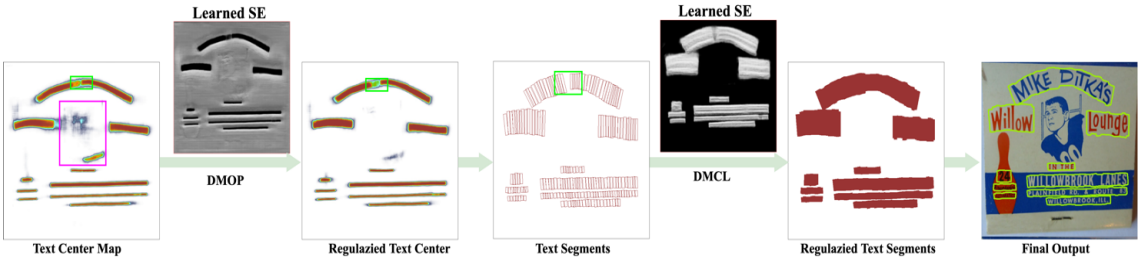
Figure 4.1 : Both of the GCN-based method [1] and the top-down method [2] have failed (as shown in (a) and (b)) when the text instance is separated due to heavy occlusion. With our morphology regularization, such separated text segments can still be connected into a single text instance (as shown in (c)).

training sometimes increases the difficulty of training and can be more complicated than the purely CNN-based top-down methods [2, 14, 15, 58, 86]. Furthermore, the GCN module only makes inferences on the relationship between text segments and makes no selection or rectification on the text segment candidates, so it does not discriminate whether a text segment is a true positive or false positive detection. Undiscriminated false text segments can lead to error accumulation, which affects the performance of the subsequent processing.

The second constraint of bottom-up methods is the difficulty of reasoning the connections between text segments. GCNs seem to be an ideal solution for relational reasoning compared to the previous sequence-based, rule-based, RNN-based or geometric-location-based methods in [11, 18–21], since GCNs are able to cap-



(a) The false text segment detection (pink boxes) and missing connections (green boxes) issues of the existing bottom-up approaches



(b) Morphtext

Figure 4.2 : Our proposed MorphText approach effectively addresses the two key issues that restrain the performance of the bottom-up methods. The pink boxes indicate the false detection areas accumulated from the earlier process and the green boxes indicate the disconnected areas.

ture the complex topological structure between text segments. However, GCNs also require large computational resources for building the topological structure of all text segments. GCNs adopted by the bottom-up methods are expected to address the linkage problem of spatially separated text segments, but the performance of GCN link prediction is dependent on the quality of the visual representation of text segments.

Before generating text segments, the network first needs to predict the entire candidate text area so that text segments can be generated by slicing or regression. However, the segmentation of text may not strictly align with the spatial interval of the text instance, unless the dataset provides char-level annotation. Therefore, there are those simple cases where the text segments produced by the visual representation stage are clearly spatially adjacent, as well as those hard cases, where text segments are spatially separated. Thus, on one hand, spatially adjacent text segments (simple

cases) are already connected in the original text map. However, GCNs unnecessarily compute their connectivity again, adding unnecessary computational burden. On the other hand, for spatially separated text segments (difficult cases), it is also challenging for GCNs to determine whether the separation is due to the accumulation of errors from the previous CNN stage or the text instance is indeed separated at this point (see Figure 4.1(a)). More linkage failure cases from GCN-based methods [1] are shown in Figure 4.9. This is one of the reasons that the performance of GCN-based bottom-up methods [1, 28, 34] for arbitrary shape text detection is lower than those of top-down methods [2, 13, 14, 62], since GCNs in these methods only perform link prediction while having no ability to prevent error accumulation. Therefore, with additional mechanisms re-checking false text segments, GCNs can be an ideal framework for addressing the linkage problem.

Then, is there a method that can remove false detection and address the miss connectivity problem of the bottom-up approaches simultaneously?

Before the dominance of the deep learning techniques, morphological operations [87–90] had experienced great prosperity in the areas of image processing and text detection. Morphological operations, including erosion, dilation, opening and closing, are designed to analyze the shape and form of an object in the image, so as to filter target patterns and connect distinct patterns in the image. They can be directly applied to gray scale images with a predefined Structure Element (SE). Compared to CNNs, morphological operations are non-linear with max/min selection according to the SE [91]. Moreover, the efficiency of filtering a specific pattern by a morphological operation is higher, because CNNs need to stack several layers to accomplish the same task, whereas morphological operations only need to select a proper SE. Apparently, the selection of SE plays a key role in morphological operations. However, determining the shape, size and flatness of an SE is a process of trial and error and does not always bring performance gains. Handcrafted SEs have significant limitations and cannot adapt to different patterns. This is why traditional morphological operations have gradually lost their attraction for image processing tasks.

Given the rapid development of deep learning techniques, traditional morphological operations have also been revitalized and evolved into Deep Morphological Networks (DMNs) with learnable SEs to capture different patterns in images [91,92]. These methods have also attempted to prove that DMNs can substitute CNNs as a universal feature extractor or classifier. Nevertheless, the reality shows that this is not really an area where morphology specialises, often leading to incremental improvement only. Moreover, these methods have not compared their DMNs with SOTA CNN models or tested on more challenging benchmark datasets. In our work, we propose to utilize the deep morphology as a supplementary of CNNs so as to take advantage of their strength in terms of preserving the regular patterns of targets and tackle the linkage problem that linear filters suffer.

Building on the strength of deep morphological operations in preserving patterns of regularity, in this chapter we propose a novel arbitrary-shape scene text detection approach to regularize text segments and alleviate the false detection and missing linkage problems of the existing bottom-up methods. Figure 4.2 shows such an example of handling the existing problems.

In particular, for removing false detection patterns, we design a Deep Morphological Opening (DMOP) module and embed it into the generation process of text segments and their center-line maps so as to suppress the noisy text-line interfering patterns. Note that our proposed DMOP is different from the opening operations in [91,92], where our aim is to regularize the text segment detection results, instead of being used as a feature extractor. Through our proposed embedding of the DMOP module, the core strength of the traditional morphological opening operation is revitalized with a trainable kernel to filter those text-like noisy patterns. Figures in Figure 4.2 illustrate this. Thus, our DMOP is designed to regularize the results of CNN under the guidance of our carefully designed loss functions. Moreover, we propose to adopt the residual connection from [73] to ensure our deep morphological opening retains its partial differentiable properties and does not over-process the detection results. With the DMOP module, the error-prone post-possessing steps used in the existing bottom-up methods are completely unnecessary thanks to the

trainable morphological module.

For linking different text segments after the DMOP regularization, we design a Deep Morphological Closing (DMCL) module which operates directly on the morphology of each text segment to fill the gaps between them and build the link. Here, even if the partial text segments are missing due to insufficient feature representation (*e.g.*, Figures. 4.1(a) and 4.2(b)), the DMCL module can still have a chance to determine their morphology. The DMCL makes text segments free from the unchangeable geometric form and relaxes the restriction of the existing bottom-up methods on text segments for fixed geometric forms, allowing text instances of various shapes to stretch in most significant direction.

The major contributions of this chapter are as follows.

1) Towards accurate detection of arbitrary-shape text, we propose a novel MorphText approach by effectively embedding deep morphology for regularizing text segments.

2) Our method enables the overall network to be trained in an end-to-end manner and replaces the error-prone post-processing steps in bottom-up methods with two trainable deep morphological modules.

3) Our method outperforms the state-of-the-art arbitrary-shape text detection methods on several competitive benchmark datasets. To the best of our knowledge, this is the first time that deep morphology is introduced into this area.

4.2 Related Work

4.2.1 Removing False Detection in Arbitrary-shape Text Detection

Some methods, such as those in [2, 12, 15, 18, 28, 34, 66, 67, 85], do not attempt to remove false detection explicitly. Instead, they often make their efforts on redesigning the feature fusion model by choosing a better backbone network structure or applying multi-scale training and testing strategies to enhance the networks' feature capturing ability for text. The other methods, *e.g.*, the ContourNet in [14], use the idea of bagging in ensemble learning in which text areas are determined by

more than one text map. The text centre line map is an additional text map and it often reflects the core text area and has also been adopted by [1, 11, 29, 93, 94], since it can effectively reduce the problem of overlapping text regions in text maps. Considering text areas and text centre lines together and also use the idea of bagging, the methods [1,11] performed element-wise multiplication on the text map and text centre line map. However, the idea of bagging treats text maps equally, ignoring the different quality of various text maps, leading to inaccurate results. Moreover, it is very likely that some text maps contain flaws such as missing detection of text. Consequently, the element-wise multiplication wrongly removes the true text objects.

In our proposed approach, text segments obtained from the text segment proposal network are firstly regularized by deep morphology modules. This alleviates the problem of false detection being propagated into the subsequent process. Moreover, inspired by [73], we introduce the residual connection into our network, which effectively leverages the regularization so that true text segments are less likely to be over-regularized during this process.

4.2.2 Deep Morphological Networks

Traditional morphological operations have bloomed in the past few decades for image processing tasks such as segmentation, text detection and image classification [88–90, 95, 96]. However, the selection of structure elements (SEs) and the enumerative combination of the two basic morphological operations, *i.e.*, erosion and dilation [97], are the key steps for extracting morphological features and succeeding on the elaborate tasks, requiring the knowledge of the expert. The robustness of these handcrafted steps cannot be guaranteed. Thus, the idea of learning the structure elements automatically is feasible and logical to be considered during the current deep learning era [98]. Zamora *et al.* [98] trained morphological neurons with one dimensional SE by a fully connected neural network. Recently, deep morphological methods [91, 92, 99, 100] extended the representation of SEs to two dimensions, and enabled 2D images to be the input of deep morphological networks. Among them, Nogueira *et al.* [91] used a flat SE to build morphological neurons for extracting

features as a replacement of linear convolutions. However, flat SEs only define the shape of the filtering process and do not affect the value of the pixel, so they may lead to insufficient feature representation on images. Later, trainable non-flat SEs were introduced in [100] where a morphology based pooling method was developed to replace the max pooling process in CNNs. Mondal *et al.* [92, 99] also adopted trainable non-flat SE and proved that the alternative stacking of deep morphological erosion and dilation could help with image restoration problems without needing CNNs.

Although the methods in [91, 92, 99] showed that deep morphological networks contained far fewer parameters than CNNs, it is difficult for deep morphological networks to compete with CNNs in terms of feature extraction and classification capabilities. This conclusion can be drawn if we look at the results of image dehazing, image classification and de-raining tasks in [91, 92, 99]. The min/max algebra in the morphological operation can only be piece-wise differentiable during training, resulting in the inability to precisely and progressively update the gradient at each pixel when compared with the back-propagation process in CNNs. Moreover, the min/max algebra is similar to the max pooling layer in CNNs and it inevitably results in the loss of information.

4.3 Methodology

The architecture of our method is illustrated in Figure 4.3, which contains three parts, namely, a text segment proposal module, a deep morphology based regularization module, and a morphological linking module. Text images are first fed into the text segment proposal module, which uses ResNet50 [73] as backbone and FPN [53] as neck to extract multi-scale features of texts. Then, the multi-scale features are concatenated and processed by the output head. With the output head, a text probability map and its center probability map indicating the center points of text segments, and two probability maps indicating the height and rotation angle of text segments, are generated. The DMOP module regularizes the noisy text-like patterns in the text and text centre maps with learned Structure Elements and re-

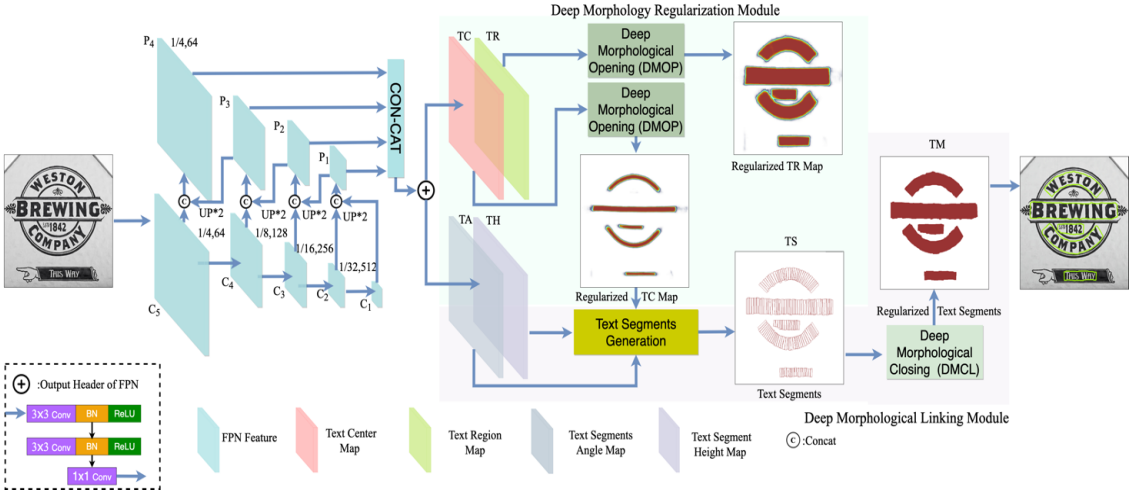


Figure 4.3 : The overall structure of our network, where "1/4,64", "1/8,128",... and "1/32,512" indicate the resize ratio and the channel number.

moves possible false text segments. The regularized text centre map is then used to guide the fusion of the text segment height and angle maps and produce candidate text segments. Finally, the resultant text segments are processed by the DMCL module, which determines the connection between them based on their morphology and produces the final result.

In the following sub-sections, we focus ourselves on detailing the two deep morphology modules and then give the details of the text segment proposal module, together with the details of the objective function.

4.3.1 Deep Morphological Operations

The deep morphological operations in our approach achieve similar functions as the traditional image morphological operations but with non-flat, trainable structure elements [97, 99, 101]. Moreover, the input of our deep morphology modules is multiple-channel image features generated from the text segment proposal module and FPN, rather than gray scale images, as indicated in Figure 4.3.

Let us denote the multi-channel image feature of size $C \times W \times H$ by $\mathcal{I}$, where C, W and H are the channel, width and height of the input image features, respectively. Similarly, let us denote the size of the structure element S by $C \times M \times N$, where C,

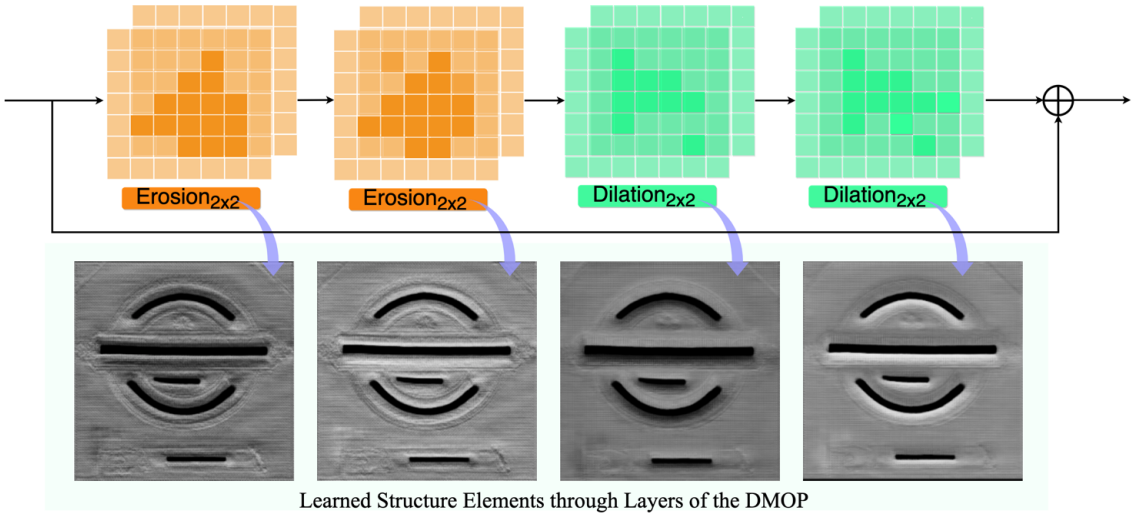


Figure 4.4 : The network structure of the DMOP module.

M and N are the channel, width and height of the structure element, respectively. The two basic image morphological operations, *i.e.*, dilation ($\oplus$) and erosion ($\ominus$), can then be expressed by the operation of the structure element S on the input I as:

$$(\mathcal{I} \oplus S)[c, w, h] = \max_{(i,j)\in D_s} (\mathcal{I}(c, w+i, h+j) + S(c, i, j)), \tag{4.1}$$

and

$$(\mathcal{I} \ominus S)[c, w, h] = \min_{(i,j)\in D_s} (\mathcal{I}(c, w+i, h+j) - S(c, i, j)), \tag{4.2}$$

where, $c \in [1, C]$, $w \in [1, W]$, $h \in [1, H]$, and D_s is the definition field of the structure element S, which can be determined by M and N. In Eqs. (4.1) and (4.2), $\mathcal{I}(c, w+i, h+j)$ means that the structure element S slides on the c-th channel of the image feature $\mathcal{I}$, in a similar manner as the convolution operation in CNNs. In our experiments, the structure elements for dilation and erosion are initialized as zero matrices, which are then learned through optimizing the objective function.

4.3.2 Deep Morphology based Text Segment Regularization

False text segments affect the accuracy of both bottom-up and top-down methods. False detection means that text-like interfering patterns are also included in the resultant text segments.

The traditional morphological opening operation [97] is an erosion operation followed by a dilation operation, and it can be used for removing small, irregular objects from detection. The basic principle of our design is also a series of deep erosion operations followed by a series of deep dilation operations. Figure 4.4 shows the structure of our Deep Morphological Opening (DMOP) module, which is composed of two erosion blocks followed by two dilation blocks. As can be seen in Figure 4.4, the learned SEs gradually suppress the noise patterns with the deep morphological operations. Here, the learned SEs are obtained by splicing the structure elements according to their positions in the image feature.

Considering an input text centre map TC, the text centre map after applying the DMOP module, denoted by TC', can be written as:

$$TC' = (TC \ominus S_1^{2\times2\times2} \ominus S_2^{2\times2\times2}) \oplus S_3^{2\times2\times2} \oplus S_4^{2\times2\times2}, \tag{4.3}$$

where $S_1^{2\times2\times2}$ to $S_4^{2\times2\times2}$ are trainable structure elements with size 2×2 of the same channels as the text centre maps. The text region map is processed by DMOP in a similar way.

Note that, a major difference between our DMOP and the deep morphological methods in [91, 92, 99, 100] is that we introduce the residual connection. Since the min/max operation in erosion and dilation are piece-wise differentiable, the gradient descent of the min/max values will be updated in SEs during the training process. For instance, if we consider L as the loss of the structure element S for Eq. (4.1), the gradient update of S can be represented by $\frac{\partial L}{\partial S}$, which can be piece-wise calculated according to the chain rule:

$$\frac{\partial L}{\partial S} = \sum_{(i,j)\in D_s} \frac{\partial(\mathcal{I} \oplus S[c,w,h])}{\partial S(c,i,j)} \cdot \frac{\partial L}{\partial(\mathcal{I} \oplus S[c,w,h])}. \tag{4.4}$$

According to the max/min algebra,

$$\frac{\partial(\mathcal{I} \oplus S[c,w,h])}{\partial S(c,i,j)} = 0 \quad or \quad 1. \tag{4.5}$$

That is to say that Eq. (4.5) equates one if and only if $\mathcal{I}(c, w+i, h+j) + S(c, i, j)$ reaches the maximum. Thus, when the expression in Eq. (4.5) is equal to zero it causes the gradient update of the SE to ignore the non-min/max values and not be constrained. Therefore, when the SE removes true positive text areas and causes over-correction, the back-propagation from non-min/max values should also be preserved to compensate for the gradient update.

In our approach, we propose to introduce a residual connection into the deep morphology regularization module, where the regularized text centre map, denoted by TC°, after applying DMOP regularization, is computed as:

$$TC^{\circ} = TC + TC', \tag{4.6}$$

where TC' is computed by Eq. (4.3).

Furthermore, different from the methods in [91, 92, 99, 100], which tried to concatenate a sequence of erosion and dilation and their dual sequence and used them as a classifier or feature extractor, our method focuses on using the deep morphological operation as a regularization module. Thus, we select small SEs and shallow deep morphological layers to constrain the influence of the DMOP module onto regularizing false detection only. Section 4.4.4 shows ablation studies validating the effectiveness of the selection of the SE size and the effect of different sizes and ways of combination.

4.3.3 Deep Morphology based Relational Reasoning

The center location of text segments TS can now be positioned using TC°. The other geometric features of text segments are described in Sect. 4.3.4. After the text segments are obtained, some of the existing bottom-up methods try to firstly determine the linkage relationship between them and then the visiting order of their contour points. However, if we look at the text segments in Figure 4.3, their shapes and contours are already very close to the ground truth text areas. The only problem that we need to solve is if the intervals between text segments should be retained and if the holes in each text segment should be filled up.

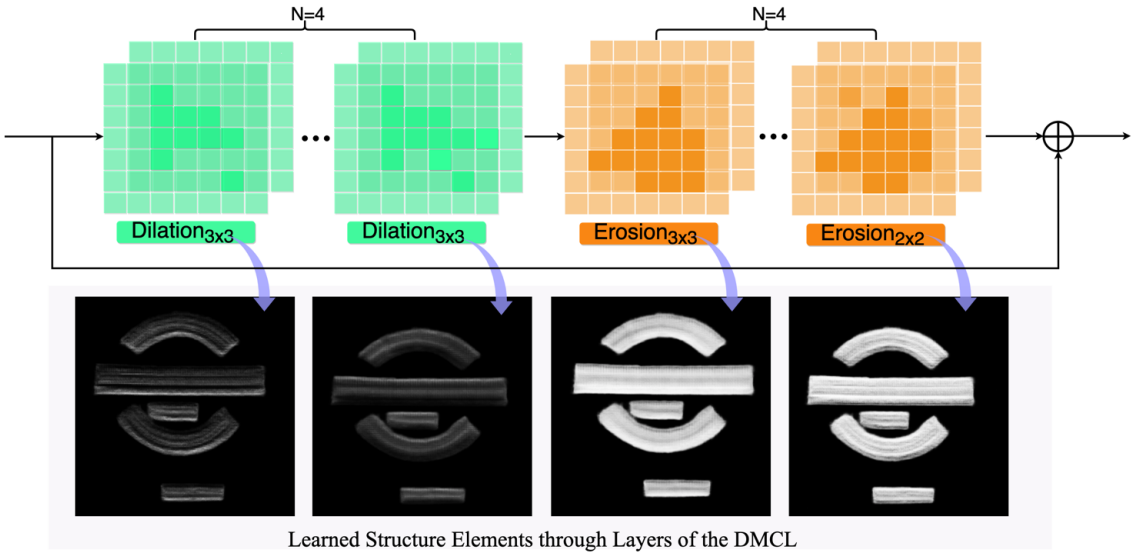


Figure 4.5 : The network structure of the DMCL module.

The traditional morphological closing operation [97], a dilation operation followed by an erosion operation, has been used to fill small holes and connect some patterns in images. Similar to the DMOP, we propose a Deep Morphological Closing (DMCL) module, which uses four deep morphological dilation operations followed by four erosion operations to build the link between text segments and fill the holes in them (see Figure 4.5). Similarly, to avoid the over-correction problem, the residual connection is also adopted in our DMCL module.

Thus, the regularized text segment map after applying the DMCL module can be represented by:

$$TS^{\bullet} = TS + TS'', \tag{4.7}$$

where

$$TS' = TS \oplus S_1^{1\times3\times3} \oplus S_2^{1\times3\times3} \oplus S_3^{1\times3\times3} \oplus S_4^{1\times3\times3}, \tag{4.8}$$

and

$$TS'' = TS' \ominus S_5^{1\times3\times3} \ominus S_6^{1\times3\times3} \ominus S_7^{1\times3\times3} \ominus S_8^{1\times3\times3}. \tag{4.9}$$

Here, the size of the SEs is set to 3×3 empirically. The stacking of four deep morphological layers is also determined empirically to fill small holes in the text segment maps. The learned SEs in Figure 4.5 show the process of text segments

being reshaped and filled to a smooth text instance.

After applying the DMCL module, the text detection results can be directly found in the text segment maps with the hole and connection problem fixed. Thus, our DMCL introduces trainable SEs to build the link and fill the holes, and to replace the tedious post-processing steps, such as those in methods [1, 14, 28, 34]. Our method does not need to locate the contour points or identify their visiting order. The contour drawing process now is the same as in top-down methods. Thus, post-processing steps such as segment grouping and false detection removal are packed into an end-to-end manner.

4.3.4 Text Segment Proposal Module

The proposed text segment proposals in our work are rectangles with rotation angles. Each text segment can be represented by $[x, y, h, w, \theta]$, namely, the x and y coordinates of the center points, the height, width and rotation angle of the text segment. Here, we concatenate the FPN features $P1 - P4$ and use an output head (see Figure 4.3) to obtain the 6-channel geometric features GF of the text segments, where $GF \in \mathbb{R}^{6\times W\times H}$. The GF contains four text probability maps, namely, two channels for the Text Region (TR) map, two channels for the Text Center (TC) map, one channel for the Text Height (TH) map and one channel for Text Angle (TA) map, *i.e.*, TR$\in \mathbb{R}^{2\times W\times H}$, TC$\in \mathbb{R}^{2\times W\times H}$, TH$\in \mathbb{R}^{1\times W\times H}$ and TA$\in \mathbb{R}^{1\times W\times H}$. Among them, the TR map is not directly involved in the calculation of text segments but is used to guide the training of the FPN feature extractor. The ground truth of TC maps is generated by shrinking the polygon using the clipping algorithm proposed in [102], where the shrinking ratio is set to 0.2.

To generate the ground truth of TH and TA maps, we first use the method in [74] to find the bottom long side of the polygon representation of each text region. Then, the bottom long side is sectioned to n line segments with $n - 1$ points. Here, for every four pixels we sample a point. To generate the TH map, for each point in the TC map, we find the distance from this point to its closest line segment and two times the distance is used to approximate the height of this point. Then, we

calculate the angle of the distance line of each point in the TC map with a horizontal line to obtain the TA map.

Note that GF does not produce the feature map for the width of the text segments. This is because our DMOP module has trainable SEs, so that it is not necessary to predict the exact width of the text segments, as each text segment will go through the dilatation operation in the DMOP module. Thus, in our approach, we simply calculate the Text Width (TW) map based on the resultant TH map. Moreover, we design the text segments as slender rectangles, mimicking the general shape of characters. Thus, the TW map is calculated by:

$$TW = Clip(2, \frac{1}{4}TH, 8), \tag{4.10}$$

where $Clip$ is the clipping function that limits the values in the TW map to the range of $2 - 8$ pixels.

Then, with TC, TH, TW and TA, the TS can be generated for the DMOP module. For every point in the TC map, a rectangle text segment is generated where text segments with over 50% overlapping are removed by non-maximum suppression to reduce unnecessary computation.

4.3.5 Objective Function

The overall objective function consists of five parts:

$$L = \lambda_1 L_{TR} + \lambda_2 L_{TC} + \lambda_3 L_{TH} + \lambda_4 L_{TA} + \lambda_5 L_{TM}, \tag{4.11}$$

where L_{TR} and L_{TC} are the losses from the regularized TR and TC maps, and L_{TM} is the loss from text segment after morphological operation. They are all class balanced cross-entropy loss for guiding the training of text areas and their center line areas.

The OHEM [77] is adopted for training the L_{TR} and L_{TM} keeping the ratio of positive sample number and negative sample number to $1 : 3$. L_{TC} is also a class-balanced cross-entropy loss with the balancing factor between positive samples and

negative samples being set to 0.75 empirically to address the unbalanced-sample problem between the text pixels and background pixels. L_H is the loss for the height of the each text segment, calculated by:

$$L_{TH} = SmoothedL1(h, \hat{h}), \tag{4.12}$$

where h and $\hat{h}$ are the ground truth height and the predicted height of text segments.

L_{TA} is the angle loss of text segments, calculated by:

$$L_{TA} = \frac{1}{N}\sum_{i \in N}(1 - cos(\theta_i, \hat{\theta}_i)), \tag{4.13}$$

where θ and $\hat{\theta}$ are the ground truth and the predicted value of the rotation angle of the text segments, and N is the total number of text segments. In this chapter, λ_1, λ_2, λ_3, λ_4 and λ_5 are set to 1, 2, 1, 1 and 1, respectively, for simplicity.

4.4 Experiments

In this section, quantitative analysis is conducted to evaluate the performance of our method. Comprehensive experiments are conducted on four widely-used scene text detection benchmarks, where the effectiveness on arbitrary-shaped text detection is verified on Total-Text [9] and CTW1500 [10], the effectiveness on quadrilateral text detection is tested on MSRA-TD500 [79] and the effectiveness on multi-lingual text detection is verified on ICDAR2017-MLT [80].

Note that the selection of the backbone of the network as well as training tricks such as the multi-scale training and testing significantly influences the detection results [33]. Moreover, there are different ways of multi-scale training and testing used in practice, making it hard to compare their performance. To avoid the impact of different backbone structures and multi-scale training and testing, we use ResNet50 as the feature extraction backbone of FPN and adopt only single-scale training and testing in our experiments.

In the following sub-sections, we first introduce the datasets tested in the ex-

periments and the implementation details, and then, quantitative and qualitative comparisons are made between our MorphText and the state-of-the-art approaches. Ablation studies are presented to show the effectiveness of the proposed modules and the impact of the parameter setting.

4.4.1 Implementation Details

Our MorphText is implemented using the PyTorch 1.7 framework. The backbone network ResNet50 is pre-trained on ImageNet, with FPN being adopted for multi-scale feature extraction. The channels of the up-sampling pipeline are set to 512, 256, 128 and 64 respectively.

The training of MorphText can be divided into two steps. First, we pre-trained our model on SynthText for five epochs using the Adam optimizer with a learning rate of 0.001. The input image size for pre-training was set to 640×640 pixels. Then, before we fine-tuned our model on specific datasets, ICDAR2017-MLT was used as an additional pre-train dataset for fine-tuning for another 100 epochs. We adopted the SGD optimizer and an initial learning rate of 0.01, which was decayed by a factor 0.1 for every 100 epochs. The momentum and weight decay of SGD were set to 0.9 and 0.0001, respectively. Following the existing practice, the input size was set to 640×640 for CTW1500 and Total-Text and 960×960 for MSRA-TD500 and ICDAR2017. The numbers of fine-tuning epochs for CTW1500, Total-Text, MSRA-TD500 and ICDAR2017 were set to 200, 200, 250, and 400, respectively. Data augmentation techniques including random cropping, resizing, colour variations, adding random noise, flipping and rotation were adopted.

During the testing, following the existing practice, the input images were resized to 640×640, 800×800, 1024×1024 and 1024×1024 for CTW1500, Total-Text, MSRA-TD500 and ICDAR2017-MLT, respectively. The training was conducted on a single NVIDIA RTX 3090 GPU and the testing was conducted on a single NVIDIA Quadro P6000 GPU with a 3.60GHz Intel Xeon Gold 5122 CPU.

Figure 4.6 : Examples of text detection results obtained with the proposed MorphText approach on benchmark datasets. The first three rows are the results obtained from arbitrary-shape text detection datasets CTW1500 and Total-Text. The last row shows the results obtained from multi-oriented datasets MSRA-TD500 and ICDAR2017-MLT.

4.4.2 Inference

During the inference stage, only TC, TH and TA maps are generated from FPN. The TW map is calculated from Eq. (4.10). With the geometric features, the text segments are drawn on a blank image. Here, NMS with a threshold of 0.5 is applied to remove highly overlapping text segments. Then, the DMCL module is used to regularize text segments, and the contour finding process can be directly implemented on this. Our method does not need to locate the contour points or identify their visiting order. The contour drawing process now is the same as the top-down methods. Thus, post-processing such as segment grouping and false detection removing

is packed into an end-to-end manner.

4.4.3 Comparison on Benchmark Datasets

In this section, we present the experimental results conducted on Total-Text, CTW1500, MSRA-TD500 and ICDAR2017-MLT datasets. Some of the detection results of our proposed method are visualized in Figure 4.6.

For fair comparison, networks with appropriate volumes such as VGG16 and ResNet50 are selected as the feature extraction backbones to exclude performance gain brought by the improvement of the backbones. Moreover, single-scale testing results are recorded to exclude performance gain brought by multi-scale testing.

Tables 4.1 to 4.4 compare the detection results of MorphText with those of the state-of-the-art approaches on the four benchmark datasets, respectively. The "-" in the tables indicates that the comparative method has not reported their results on the dataset. The "EXT" in the tables indicates the comparative method has used external data for pre-train.

Comparison on CTW1500: As shown in Table 4.1, we first compare our results with the SOTA methods on CTW1500. Our method achieved **90.0%**, **83.3%** and **86.5%** in precision, recall and F-measure, and outperforms all of the comparison methods. Compared with the pioneering bottom-up methods [1, 28], our model has improved the detection accuracy by 1.7 percent. This result demonstrates that our DMCL module can replace the complex GCN reasoning process to make an inference on the relationship between text segments. When the DMOP module is applied to suppress false detection, the precision of our method has been significantly improved to 90.0% on CTW1500. Our method remedies the error-prone post-processing steps in the bottom-up methods. Compared with the pioneering top-down method [13], our method has improved the F-measure by 1.0 percent on the CTW1500. The DMOP module introduces a further rectification under the guidance of the loss function, and it provides another way to remove false detection instead of introducing deformable or non-local convolutional blocks, as in the

Table 4.1 : Results on CTW1500. (§: top-down methods, †: bottom-up methods)

Method	Venue	Backbone	EXT	P (%)	R (%)	F (%)
ATRR § [12]	CVPR'19	VGG16	×	80.1	80.2	80.1
LOMO§ [29]	CVPR'19	ResNet50	✓	85.7	76.5	80.8
DB§ [32]	AAAI'20	ResNet50	✓	86.9	80.2	83.4
SDASSC § [59]	TMM'21	ResNet50	×	86.2	80.4	83.2
MS-CAFA § [60]	TMM'21	ResNet50	×	85.7	85.1	85.4
Mask TTD § [103]	TIP'20	ResNet50	×	79.7	79.0	79.4
ContourNet§ [14]	CVPR'20	ResNet50	×	85.7	84.0	84.8
TextPerceptron§ [61]	AAAI'20	ResNet50	✓	87.5	81.9	84.6
TextFuseNet§ [2]	IJCAI'20	ResNet50	✓	85.0	**85.8**	85.4
PCR § [62]	CVPR'21	DLA34 [63]	✓	87.2	82.3	84.7
FCENet§ [13]	CVPR'21	ResNet50	×	87.6	83.4	85.5
TextSnake† [11]	ECCV'18	VGG16	✓	67.9	85.3	75.6
PSENet† [65]	CVPR'19	ResNet50	✓	84.8	79.7	82.2
TextRay† [21]	MM'20	ResNet50	×	82.8	80.4	81.6
OPOM† [66]	TMM'20	ResNet50	✓	85.1	80.8	82.9
CRAFT† [18]	CVPR'19	VGG16	✓	86.0	81.1	83.5
TextDragon† [35]	ICCV'19	VGG16	✓	84.5	82.8	83.6
PuzzleNet† [34]	arXiv'20	ResNet50	✓	84.1	84.7	84.4
DRRG† [1]	CVPR'20	VGG16	✓	85.9	83.0	84.4
ReLaText† [28]	PR'21	ResNet50	✓	86.2	83.3	84.8
Ours†	-	ResNet50	×	89.0	83.2	86.0
*Ours†	-	ResNet50	✓	**90.0**	83.3	**86.5**

methods shown in [13,28,32,59,66]. Thanks to our proposed bottom-up method embedded with the deep morphological blocks, the results on CTW1500 demonstrate the superb effectiveness of MorphText when dealing with long curved texts.

Comparison on Total-Text: As shown in Table 4.2, our method achieves **90.6%**, **85.2%** and **87.8%** in precision, recall and F-measure on Total-Text, and outperforms the state-of-the-art methods [13] by 2.0 percent in F-measure. The DMCL module can also handle short curved texts in Total-Text flexibly, namely, breaking the text instance when it is appropriate to do so according to the loss.

Figure 4.7 qualitatively compare the visual detection results of the proposed MorphText with some recent top-down and bottom-up methods, *i.e.*, TextFuseNet [2] and DRRG [1], obtained on some samples containing long and curved text instances.

Table 4.2 : Results on Total-Text. (§: top-down methods, †: bottom-up methods)

Method	Venue	Backbone	EXT	P (%)	R (%)	F (%)
ATRR § [12]	CVPR'19	VGG16	×	80.9	76.2	78.5
LOMO § [29]	CVPR'19	ResNet50	✓	88.6	75.7	81.6
DB § [32]	AAAI'20	ResNet50	✓	87.1	82.5	84.7
SDASSC § [59]	TMM'21	ResNet50	×	85.4	81.2	83.2
MS-CAFA § [60]	TMM'21	ResNet50	✓	84.6	78.6	81.5
Mask TTD § [103]	TIP'20	ResNet50	×	79.1	74.5	76.7
ContourNet § [14]	CVPR'20	ResNet50	×	86.9	83.9	85.4
TextPerceptron § [61]	AAAI'20	ResNet50	✓	88.8	81.8	85.2
TextFuseNet § [2]	IJCAI'20	ResNet50	✓	87.5	83.2	85.3
PCR § [62]	CVPR'21	DLA34 [63]	✓	88.5	82.0	85.2
FCENet § [13]	CVPR'21	ResNet50	×	89.3	82.5	85.8
TextSnake † [11]	ECCV'18	VGG16	✓	82.7	74.5	78.4
PSENet † [65]	CVPR'19	ResNet50	✓	84.0	78.0	80.9
TextRay † [21]	MM'20	ResNet50	×	83.5	77.8	80.6
OPOM † [66]	TMM'20	ResNet50	✓	88.5	82.9	85.6
CRAFT † [18]	CVPR'19	VGG16	✓	87.6	79.9	83.6
TextDragon † [35]	ICCV'19	VGG16	✓	85.6	75.7	80.3
DRRG † [1]	CVPR'20	VGG16	✓	86.5	84.9	85.7
ReLaText † [28]	PR'21	ResNet50	✓	84.8	83.1	84.0
Ours †	-	ResNet50	×	88.4	**85.5**	86.9
*Ours †	-	ResNet50	✓	**90.6**	85.2	**87.8**

As shown in this comparison, our method exhibits greater robustness to interfering text-like noises and is less prone to the connection problem thanks to the proposed DMOP and DMCL modules. Our results obtained on these two most representative arbitrary-shape text detection datasets demonstrate that the proposed deep morphological modules can handle the two important issues in bottom-up methods in an end-to-end manner, *i.e.*, the unstable post-processing step and the missing connections. Our proposed DMOP module is able to remove the text-like objects from text segments, making DMCL's relational reasoning more reliable.

Comparison on MSRA-TD500: On the dataset MSRA-TD500, as shown in Table 4.3, our method has also achieved a comparable SOTA result of an F-measure of 87.0%, compared with the best-performing method PCR [62]. But different from [62] that has adopted the latest backbone [63], our MorphText only utilizes the origi-

Table 4.3 : Results on MSRA-TD500. (§: top-down methods, †: bottom-up methods)

Method	Venue	Backbone	EXT	P (%)	R (%)	F (%)
ATRR § [12]	CVPR'19	VGG16	×	85.2	82.1	83.6
DB§ [32]	AAAI'20	ResNet50	✓	**91.5**	79.2	84.9
Mask TTD § [103]	TIP'20	ResNet50	✓	85.7	81.1	83.3
PCR § [62]	CVPR'21	DLA34 [63]	✓	90.8	83.5	87.0
TextSnake† [11]	ECCV'18	VGG16	✓	83.2	73.9	78.3
OPOM† [66]	TMM'20	ResNet50	✓	86.0	83.4	84.7
CRAFT† [18]	CVPR'19	VGG16	✓	88.2	78.2	82.9
PuzzleNet† [34]	arXiv'20	ResNet50	✓	88.2	83.5	85.8
DRRG† [1]	CVPR'20	VGG16	✓	88.1	82.3	85.1
ReLaText† [28]	PR'21	ResNet50	✓	90.5	83.2	86.7
Ours†	-	ResNet50	×	88.5	82.7	85.5
*Ours†	-	ResNet50	✓	90.7	83.5	**87.0**

Table 4.4 : Results on ICDAR2017-MLT. (§: top-down methods, †: bottom-up methods)

Method	Venue	Backbone	EXT	P (%)	R (%)	F (%)
LOMO§ [29]	CVPR'19	ResNet50	✓	78.8	60.6	68.5
DB§ [32]	AAAI'20	ResNet50	✓	**83.1**	67.9	74.7
SDASSC § [59]	TMM'21	ResNet50	×	79.5	66.8	72.6
PSENet† [65]	CVPR'19	ResNet50	✓	73.7	68.2	70.9
OPOM† [66]	TMM'20	ResNet50	✓	82.9	70.5	76.2
CRAFT† [18]	CVPR'19	VGG16	✓	80.6	68.2	73.9
DRRG† [1]	CVPR'20	VGG16	✓	74.5	61.0	67.3
Ours†	-	ResNet50	×	81.9	74.0	77.8
*Ours†	-	ResNet50	✓	82.8	**74.2**	**78.3**

nal ResNet50 as the feature extraction backbone. Moreover, the modeling of our MorphText follows the regularity of texts, using the DMCL module to connect character-like text segments instead of introducing complex feature fusion process. Furthermore, different from the methods [28, 34] that used multi-scale training, our method adopts a fixed scale to train the model. The results on MSRA-TD500 prove that the DMCL module can also link non-Latin text segments, allowing the MorphText to deal with long multi-oriented texts in an end-to-end manner.

Comparison on ICDAR2017-MLT: The text detection results on the multi-

(a) Ours (b) TextFuseNet [2] (c) DRRG [1] (d) Ground truth

Figure 4.7 : Qualitative comparisons with the SOTA methods on challenging samples.

lingual and multi-oriented dataset ICDAR2017-MLT are shown in Table 4.4. As it shows, our MorphText has achieved **82.8%**, **74.2%** and **78.3%** in precision, recall and F-measure, so it also surpasses the best performing method [66] by 2.1 percent in F-measure. Text segments of non-Latin languages such as Chinese and Japanese are different to those of English, where non-Latin languages have no space between words. Therefore, connecting non-Latin text segments requires additional linkage information, since the intervals between non-Latin text segments vary largely. Methods such as those in [1,2,18,28] attempted to introduce a char-level annotation as well as GCN to guarantee the connectivity. Different from these methods, our method can directly regularize text segments by the proposed deep morphological modules, so the multi-lingual text segments are less likely to affect the performance. The highest F-measure and recall rate show the robustness of our approach in detecting multi-lingual texts.

4.4.4 Ablation Studies

The Effectiveness of DMOP and DMCL

Table 4.5 : The effectiveness of our proposed DMOP and DMCL on CTW1500 and Total-Text.

Datasets	OP	CL	DMOP	DMCL	P (%)	R (%)	F (%)
CTW1500	×	✓	×	×	85.7	82.7	84.2
	✓	✓	×	×	86.5	82.9	84.7
	×	✓	✓	×	87.2	83.0	85.0
	×	×	×	✓	87.6	**83.3**	85.4
	✓	×	×	✓	88.0	83.2	85.5
	×	×	✓	✓	**89.0**	83.2	**86.0**
Total-Text	×	✓	×	×	86.5	84.8	85.6
	✓	✓	×	×	87.1	85.2	86.1
	×	✓	✓	×	87.9	84.8	86.3
	×	×	×	✓	87.4	85.5	86.4
	✓	×	×	✓	87.7	85.5	86.6
	×	×	✓	✓	**88.4**	**85.5**	**86.9**

To validate the effectiveness of our proposed DMOP and DMCL modules, we conduct ablation studies on the two most representative datasets of arbitrary-shape text detection, *i.e.*, CTW1500 and Total-Text. Table 4.5 shows the comparison results. In this table, "P", "R" and "F" represent Precision, Recall, and F-measure, respectively, "DMOP" and "DMCL" denote our DMOP false detection removal branch and DMCL relational reasoning branch, respectively, and 'OP' and 'CL' denote the traditional morphological opening and closing, where the kernel size of OP and CL are 2×2 and 3×3, respectively, the same as those in DMOP and DMCL. Here, the baseline is the model where our proposed DMCL is substituted with the traditional closing operation but without the DMOP for noise suppression. As our method follows a bottom-up design, a closing module, either CL or DMCL, needs to present to ensure the connectivity of the text segments.

From Table 4.5, the results show that our idea of using morphological operations to remove false detection and connect separated text segments are effective. Although the SEs and iteration selection of OP and CL are based on trial and error, the OP and CL adhere the design concept of our bottom-up design and can

bring 0.5% and 0.5% gains of F-measure on CTW1500 and Total-Text, respectively. When OP and CL are replaced by our proposed DMOP and DMCL, more significant improvements have been achieved on both tested datasets by 1.8% and 1.3%, respectively. We attribute this performance gain to the guidance of the loss functions, which allows the DMOP and DMCL to be flexible and robust. To be specific, when CL is replaced by DMCL, the designing idea of using regularized text segments to fill the gap has become trainable, and it brings 1.2% and 0.8% gains on CTW1500 and Total-Text, respectively.

The purpose of OP and DMOP is to remove the false detection (text-like objects), which clears the way for the subsequent relational reasoning. This conclusion can be seen from the result of the combination of OP and DMCL, where even the traditional OP can remove some of the small text-like objects. The selection of the kernel of OP is a flat all-one matrix with a size of 2×2. There is a performance gain of 1.3% and 1.0% on both datasets. When OP is replaced by DMOP, the combination of DMOP and CL can also bring 0.8% and 0.7% gains on both datasets. The selection of the SE for CL here is a flat all-one matrix with a size of 3×3. The DMOP contains trainable kernels and can decide whether to filter according to different situations of text instances. From the results shown in Table 4.5, we can conclude that our DMOP and DMCL modules are two complementary modules. Through the gradient update process of MorphText, both DMOP and DMCL complement each other to boost the final performance.

The Impact of the SE Size

In traditional morphological operations, the selection of SEs can greatly affect the final results. Here, we mainly test the impact of the size of SEs on the performance, since the values and patterns in the SE are learned with the neural network. In our experiments, the SEs of DMOP and DMCL were initialized with all-zero matrices. We chose the grid search strategy to search the optimal kernel size for DMOP and DMCL. The possible kernel sizes of DMOP were set to $[2 \times 2, 3 \times 3, 4 \times 4, 5 \times 5]$ and the possible kernel sizes of DMCL were set to $[2 \times 2, 3 \times 3, 4 \times 4, 5 \times 5]$. We repeated each combination for five times and report descriptive statistics such as max, min,

Table 4.6 : The impact of the kernel size in DMOP and DMCL on F-measure. The F_{min}, F_{max}, F_{mean} and $F_{std-dev}$ the min, max, mean, and standard deviation for the F-measure.

Kernel Size	CTW1500				Total-Text			
DMOP, DMCL	F_{min}	F_{max}	F_{mean}	$F_{std-dev}$	F_{min}	F_{max}	F_{mean}	$F_{std-dev}$
$2\times2, 2\times2$	82.4	83.1	82.6	0.24	82.0	82.2	82.1	0.09
$2\times2, 3\times3$	85.6	86.0	85.9	0.16	86.6	86.9	86.7	0.10
$2\times2, 4\times4$	79.5	80.2	79.7	0.26	82.0	82.7	82.4	0.26
$2\times2, 5\times5$	75.6	75.8	75.7	0.08	75.6	76.0	75.7	0.14
$3\times3, 2\times2$	82.0	82.5	82.3	0.17	82.0	82.3	82.1	0.12
$3\times3, 3\times3$	85.7	85.9	85.8	0.09	86.0	86.4	86.2	0.16
$3\times3, 4\times4$	79.5	79.8	79.6	0.12	81.5	81.8	81.6	0.11
$3\times3, 5\times5$	75.2	75.4	75.3	0.08	75.3	75.6	75.5	0.10
$4\times4, 2\times2$	81.8	82.2	82.0	0.16	81.2	81.8	81.5	0.20
$4\times4, 3\times3$	83.1	83.5	83.3	0.15	82.4	82.7	82.5	0.13
$4\times4, 4\times4$	79.0	79.5	79.3	0.17	81.4	81.7	81.5	0.12
$4\times4, 5\times5$	75.0	75.4	75.3	0.15	75.3	75.7	75.4	0.14
$5\times5, 2\times2$	78.2	78.8	78.5	0.20	77.2	77.8	77.6	0.21
$5\times5, 3\times3$	82.1	82.4	82.3	0.11	82.0	82.4	82.2	0.14
$5\times5, 4\times4$	77.1	77.7	77.4	0.19	79.7	80.5	80.2	0.27
$5\times5, 5\times5$	74.1	74.7	74.4	0.23	74.7	75.0	74.9	0.11

mean and *std* in Table 4.6. The results show that the SE sizes 2×2 and 3×3 have achieved the highest average F-measure of 85.9% and 86.7% on CTW1500 and Total-Text.

Moreover, if we keep the SE size of DMOP to be 2×2, the performance decreases as the SE size of DMCL increases from 3×3. Interpreting the results, we can see that the F-measure is sensitive to the kernel size: for DMCL, when the kernel size is greater than or equal to 4×4, it may cause very close text instances to stick together and hence results in a fluctuation of F-measure on CTW1500 and Total-Text. For kernel sizes less than 3×3, sometimes the DMCL fails to deal with large gaps and holes. For DMOP, when the kernel size is equal to or larger than 5×5, over-correction occurs, also resulting in a decrease in performance. When the kernel size is larger than or equal to 5×5, it is more likely to use the DMOP as a feature extractor and have more significant impact on rectifying the former CNN features.

The results in [91, 92, 99, 100] show that when deep morphological operations are

used as the classifier or feature extractor, the SE size is usually greater than or equal to 5×5. Instead, in our approach, DMCL and DMOP are used to complement CNNs, and they work well for removing false detections, filling hole areas and smoothing the results of CNNs. A large SE tends to over-correct the detection results and is ineffective when filtering some local false detections.

The Impact of the Residual Connection

To discuss the influence of residual connection, we keep the kernel size and layers of DMOP and DMCL. Table 4.7 shows that when residual connection is removed, all of precision, recall and F-measure have decreased. This supports the claim that residual connection can alleviate the problem of the over-correction.

Table 4.7 : Comparison of the detection results with/without the residual connection

Residual	CTW1500			Total-Text		
	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
×	88.2	81.6	84.8	87.2	82.7	84.9
✓	**89.0**[+0.8]	**83.2**[+1.6]	**86.0**[+1.2]	**88.4**[+1.2]	**85.5**[+2.8]	**86.9**[+2.0]

In our proposed MorthText approach, for every point in the TC map, a rectangular text segment centered at the point is generated where neighbouring text segments with overlapping over a certain threshold are removed by NMS to reduce unnecessary computation. Apparently, this threshold affects the density of the resultant text segments. Generally speaking, the larger the overlapping, the denser are the resultant text segments and the easier it is for the DMCL module to perform relational reasoning. However, denser text segments involves higher computation. Smaller thresholds, on the other hand, will result in less text segments and faster processing, but may require additional morphological operations in DMCL.

The Impact of the NMS Threshold

In order to assess the impact of the NMS threshold on the final performance, we conduct an additional ablation study on CTW1500 and Total-Text datasets. Table 4.8 shows the results. As it can be seen from the table, our approach has

Table 4.8 : Comparison of detection results with different NMS thresholds.

Threshold	CTW1500			Total-Text		
	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
NMS@0.2	79.8	81.7	80.7	83.2	79.6	81.4
NMS@0.3	87.4	80.7	83.9	87.7	83.0	85.3
NMS@0.4	86.8	81.2	83.9	88.0	85.2	86.6
NMS@0.5	89.0	83.2	**86.0**	88.4	**85.5**	**86.9**
NMS@0.6	90.1	82.3	86.0	88.4	85.4	86.9
NMS@0.7	88.7	**83.3**	85.9	**88.6**	85.2	86.9
NMS@0.8	**90.2**	82.0	85.9	88.3	85.4	86.8

achieved the highest F-measure of 86.0% and 86.9% on the CTW1500 and Total-Text, respectively, at NMS@0.5. Although when the threshold of NMS is greater or equal to 0.5, there is no significant difference for the detection results on CTW1500 and Total-Text, the inference speed of NMS drops significantly. Therefore, in our experiments, in order to balance the inference speed and the detection accuracy, we choose NMS@0.5. Furthermore, NMS@0.5 also means that all text segments have an appropriate overlapping with each other, which can be seen as another level of connection.

The Impact of the DMOP on Top-down Methods

Table 4.9 : Comparison of detection results with DMOP on top-down methods

DMOP	CTW1500			Total-Text		
	P (%)	R (%)	F (%)	P (%)	R (%)	F (%)
×	78.8	77.3	78.0	80.2	78.7	79.4
✓	80.2	78.1	**79.1**[**+1.1**]	80.7	79.8	**80.2**[**+0.8**]

To demonstrate the effectiveness of the proposed DMOP on top-down text detection approaches, we keep the network structure of FPN and output head in the overall structure, while only predicting the TR map by removing the other parts. The function *cv2.drawContours* is directly applied to the TR map to obtain the final text detection results. The results are shown in Table 4.9.

From the results it can be seen that, the DMOP brings 1.1% and 0.8% gain on

CTW1500 and Total-Text, respectively. This shows that the DMOP module is also effective in a top-down design, similar to the bottom-up methods. This conclusion is easy to understand since the DMOP module is designed to remove the text-like interfering objects in the TR map, which are also prevalent in the top-down methods.

Time Efficiency

Table 4.10 : Inference speeds of the proposed approach on input images of different sizes in different datasets.

Datasets	Dimension	Inference Time	NMS Time	Total Time
CTW1500	640	0.12s	0.38s	0.50s
Total-Text	800	0.17s	0.45s	0.62s
MSRA-TD500	1024	0.27s	0.57s	0.84s
ICDAR2017-MLT	1024	0.31s	0.56s	0.87s

The efficiency of the proposed MorphText tested on different datasets of different input sizes is shown in Table 4.10. The figures are obtained on a workstation with a single NVIDIA Quadro P6000 GPU with an Intel(R) Xeon(R) Gold 5122 CPU.

The time consumption of our approach comes mainly from two parts. The first part is contributed by the operations of FPN and the deep morphological operations. The second part is contributed by the NMS operation. The figures in this table show that, our model runs at a decent speed. Moreover, the NMS process contributes most (64% to 76%) of the delay, irrespective of the input sizes. This could indicate a possible direction of the future research.

Discussion on Regularizing False Detections

We further validate the robustness of our method in addressing false defections. In Figures. 4.7, 4.8 and 4.9, we also show some qualitative comparison of the intermediate results and how MorphText addresses in addressing the varied sizes of noises and large linkage problem. The visualization of the results show that the proposed kernel sizes of DMOP and DMCL are robust to different types of noises and relatively large gaps between text segments. Moreover, the various interfering text-like noise patterns, as well as the large holes or gaps between text segments

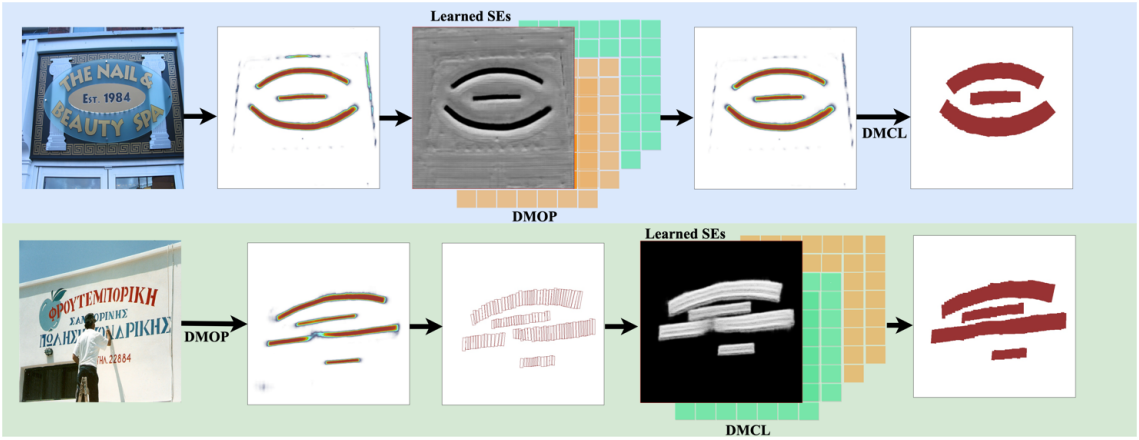


Figure 4.8 : Visualisation of the intermediate results of MorphText addressing noise patterns (top) and the connection problem between text segments (bottom).

Figure 4.9 : Qualitative comparisons with the SOTA bottom-up method [1] (top row) on handling varied sizes of interfering patterns and relative large gaps

that cause missing connection, are the main causes of false detection. The results in Table 4.11 show the effectiveness and robustness of our proposed DMOP and DMCL with kernel sizes of 2×2 and 3×3 for dealing with FPs and FNs. The DMOP/DMCL can reduce the false positive detections and address the connection problem between text segments, since both FPs and FNs decreased. Moreover, when the kernel size of DMOP/DMCL increases, they may have a negative impact on dealing with FPs and FNs.

Table 4.11 : The effectiveness of the proposed DMOP/DMCL mechanism on suppressing false positives and false negatives.

DMOP/DMCL	TPs	FPs	FNs	P (%)	R (%)	F (%)
Without	2537	423	531	85.7	82.7	84.2
$2 \times 2, 3 \times 3$	**2553**↑	**315**↓	**515**↓	**89.0**↑	83.2↑	**86.0**↑
$2 \times 2, 4 \times 4$	2429↓	614↑	639↑	79.8↓	79.2↓	79.5↓
$2 \times 2, 5 \times 5$	2313↓	706↑	755↑	76.6↓	75.4↓	75.6↓
$3 \times 3, 3 \times 3$	2571↑	344↓	497↓	88.2↑	**83.8**↑	85.9↑
$4 \times 4, 3 \times 3$	2497↓	420↓	571↑	85.6↓	81.4↓	83.4↓
$5 \times 5, 3 \times 3$	2460↓	444↑	608↑	84.7↓	80.2↓	82.4↓

Figure 4.10 : Some failure cases of the proposed MorphText, where the green bounding boxes indicate the detected results and the red bounding boxes highlight the failure areas.

4.5 Limitations

Our method can alleviate the error accumulation problem and handle texts with larger word spaces thanks to the newly proposed DMOP and DMCL modules. Some

failure cases happen with low-contrast texts, object-like texts or extremely tiny texts such as the failure cases in the first five pictures in Figure 4.10, which is also a challenge for the SOTA methods [2, 13, 62]. For some confusing text segments such as the sixth and seventh pictures in Figure 4.10, it is very hard to determine the connectivity of text segments based purely on visual features, which is also challenging for the GCN based methods [1, 28, 34]. Similar to [59, 60], our methods may also fail when text instances have large overlapping with other texts, *e.g.*, in the eighth and ninth pictures. Since overlapped texts are very rare in the training datasets, our network may not learn the features of overlapped text.

4.6 Summary

In this chapter, we have introduced deep morphology into the field of arbitrary-shape text detection for the first time, as an effective way to tackle the error accumulation of false text segment detection and the missing connection problems that prevent bottom-up text detection approaches from achieving their great potential for handling arbitrary-shape text. Two deep morphological modules have been designed and embedded into the network to regularize the text segments based on their patterns of regularity learned through training. Extensive experiments conducted on four widely-used benchmark datasets have demonstrated the effectiveness of our proposed approach. The resulting state-of-the-art performance shows that deep morphology can be used for the challenging task of arbitrary-shape text detection and that the proposed methods have demonstrated an effective approach for the task.

Chapter 5

Semantic Navigation of Slide-based Educational Video

In Chapter 3 and Chapter 4, we introduced how we can accurately locate the text instances of arbitrary-shape in complex environments. A direct application of arbitrary-shape text detection is to help with the creation of navigation tools in video classes.

With the increasing popularity of open educational resources (OERs) in the last decades, more and more users watch online videos to gain knowledge. Notwithstanding, most educational videos only provide monotonous navigation tools and hence lack elaborating annotations, which makes locating interesting content a time-consuming task. To address this limitation, in this chapter, we propose a slide-based video navigation tool that is able to extract the hierarchical structure and semantic relationship of visual entities in videos by integrating multi-channel information. Specifically, features of visual entities are first extracted from the presentation slides by a novel deep learning framework. Then, we propose a clustering approach to extract the hierarchical relationship between visual entities and use this information to associate visual entities with their corresponding audio speech text by evaluating their semantic relationship. We present two cases of using the structured data produced by this tool to generate a multi-level table of contents and notes as additional navigation materials for learning*. The evaluation experiments demonstrate the effectiveness of our proposed solutions for visual entity extraction, hierarchical relationship extraction, as well as corresponding speech text matching. The user study also shows promising improvement in the auto-generated table of contents and notes for facilitating learning.

*A demo of our proposed navigation tool https://youtu.be/9Hy6b4w1S20

5.1 Introduction

Online learning and E-Learning platforms have changed the way of perceiving education significantly. Many top universities and educational technology companies have recorded their lectures and published them on E-Learning platforms such as Coursera and Udacity. This emerging approach makes learning flexible and controllable by offering learners access to abundant excellent lectures anytime and anywhere. By the end of 2018, 101 million students worldwide have studied through MOOC, and more than 900 universities have provided 11,400 course videos for the MOOC platform [104]. Especially during the pandemic of COVID-19, the number of students taking online learning soars dramatically. Despite of the growing number of registered users and the increasing importance of online learning, approximate ninety per cent of users are unable to complete their MOOC courses and obtain their final course certificates [105].

One of the reasons is that some course videos on the MOOC are created at the instructor's own pace, but the backgrounds and knowledge levels of students are different, making it hard for learners to follow the videos. A survey [106] found that students who have eventually obtained a course certificate spent an average of 4.4 minutes watching a 12-15 minute video and, on average, skipped twenty-two per cent of the video contents. Although customized instructional strategies should be applied to individuals according to their background for effective learning [107], some of the courses on the MOOC are pre-recorded and only provide limited learning resources, leading to a barrier to students' self-paced learning. According to [108], lacking resources such as e-notes, presentation slides, e-books *etc*, has become one of the top four challenges faced by students who use online learning during the COVID-19. It is inconvenient for students who prefer to work at their own pace, as they have to spend more time finding and processing information in the lecture recordings. Educational videos typically contain massive multi-modal information such as images and audio texts. Even if learners have access to the presentation slides, it is still a challenge for learners to locate and understand key knowledge points without instructors' explanation.

Moreover, the attention of individuals is usually limited. But the length of educational videos is usually beyond this limit. Dhawan *et al.* [109] stated that the insufficient attention of individuals had become a huge problem during online learning and teaching. One of the contributors to this phenomenon is that most educational videos are very lengthy and lack elaborating navigation so that some students become bored and feel not engaged when only lecture videos are provided.

To alleviate the issues regarding inconvenient self-paced learning and limited attention, computer scientists have made various efforts to develop more effective ways by using multimedia technologies to create user-friendly navigation tools [110–113]. These navigation tools all have an ideal expectation, *i.e.*, the learners only need to 'click' on the knowledge points that they would like to learn in the slides, and the navigation system will automatically 'retrieve' the explanation of this knowledge point from a lengthy video. However, they are unable to build such a fine-grained navigation system due to a lack of the hierarchical structure information of the visual entities (knowledge points). Let us take the text instances in presentation slides as an example. Even the state-of-the-art deep learning based scene text detection methods [2, 14, 94] can only indicate the locations of text lines but not their hierarchical structure relationships. Nevertheless, a knowledge point may spread over several lines. Thus, only locating the text lines is not enough for extracting a complete knowledge point. They need to be merged according to their hierarchical structure relationship to make the knowledge points complete. Most of the existing navigation systems do not have the capability to analyze the layout of the knowledge points, resulting in that they can only build coarse linkage between unsorted contents in the slides and explanations. A recent study also states that an effective navigation tool should contain concrete ways to help learners to quickly and conveniently locate and process information [114]. The existing navigation tools do not provide visual entity level matching with explanation, so the efficiency of the information searching process decreases. Similarly when generating some learning tools with a summarized nature, such as a Table of Contents (TOCs) and notes, the coarse matching between visual entities and explanation also has a negative impact

on the completeness and preciseness of the generated information.

To address these issues, we present a slide-based video navigation tool that can effectively extract fine-grained hierarchical structure and semantic relationship of visual entities in the video by integrating multi-channel information. Unlike the existing solutions that cannot match the corresponding explanation at the visual entity level, our method adopts a state-of-the-art deep learning based model to detect the possible modules of the visual entities. Moreover, the existing solutions have considered only the positional information and the traditional hand-selected features for visual entity extraction, resulting in easily accumulated errors and unsatisfactory results. Our solution adopts deep learning techniques to automatically depict the different features of visual entities in order to largely improve the accuracy of visual entity extraction. Furthermore, to reconstruct visual entities, we depict the hierarchical relationships between visual entities instead of relying on their location and order of occurrence only. We also integrate their visual saliency and propose a clustering based approach, where visual entities belonging to the same logical or knowledge structure are merged. Finally, for utilizing the audio information of the lecture and aiming to address the limitation of the existing time-alignment approaches, we develop an algorithm to obtain the semantic relationship between speech texts and visual entities and associate a visual entity with its corresponding descriptive speech text by evaluating their semantic similarity. The structured data produced by this tool demonstrates its effectiveness in terms of generating a multi-level table of contents and notes as additional navigation cues for efficiently locating information. The main contributions of our research are as follows.

1) We develop an accurate and robust deep-learning-based approach for visual entity extraction from educational videos according to their unique features. This module greatly improves the performance of visual entity extraction, which is demonstrated by the state-of-the-art results obtained on public slide datasets.

2) We propose a visual saliency and clustering based approach to extract the hierarchical relationship among visual entities, so that it significantly ensures the completeness and preciseness of visual entities.

3) We propose an effective approach to align visual entities with speech texts by evaluating their semantic similarity. The fine-grained matching between a visual entity and its audio explanation enables the navigation system to work well for learners to locate and process information in educational videos effectively.

The rest of this chapter is organized as follows. In Section 5.2, we discuss the existing navigation tools, slide processing, and TOCs and Note generation methods. Section 5.3 presents the details of the extraction and recognition of visual entities on slides. In Section 5.4, the hierarchical relationships between visual entities are detected. Then, in Section 5.5 we evaluate the semantic similarity as well as the time-aligning relationship between visual entities and their explanatory speech texts to segment captions at the visual entity level. Section 5.6 presents two applications using our proposed method. Finally, detailed experimental results, user study as well as conclusions are presented in Sections 5.7 and 5.8 to evaluate our method.

5.2 Related Work

5.2.1 Navigation and Annotation Tools for Slide-based Educational Videos

Since presentation slides can be seen as a refinement of the essence of the video, some existing works have chosen to build their navigation systems based on slides. Chen *et al.* [115] analyzed the audio information and extracted key terms, which were then embedded into the original slides. Zhao *et al.* presented a visual navigation system using multimodal cues that could find the audio information related to each slide and provide users with an intuitive interface [8]. ViZig [70] built anchor points that were used to indicate specific slides on the timeline of the educational videos in their navigation system. However, ViZig specified that each slide must belong to one of the six types, *i.e.*, charts, codes, diagrams, flow charts, equations and tables. This is only a slide-level classification, as there are many situations where one slide contains multiple types of information, resulting in inaccurate navigation. ConceptScape [40] provided a collaborative concept map that linked the concept in the speech text to a specific frame (slide) in the video according to time alignment. However, the concept-to-frame matching is still a coarse matching since there may

be multiple concepts (visual entities) appearing in one frame, and the learners may still get confused by which concept the instructor is actually talking about. In general, the time between the appearance of a visual entity (text, formula or graph) and the following one can be recorded, and the speech text within this duration can be considered as the explanation of the previous visual entity. Matching a visual entity and its corresponding speech text using time information is called time alignment, which, however, can only pick up speech texts for visual entities appearing sequentially or one by one. Most of the existing works can only find slide-level correspondence between visual entities and speech text, *i.e.*, they can allocate corresponding subtitles to each slide. However, when users would like to learn more specific content in the slide, they still have to go through all of the videos that correspond to the slide.

Thus, creating matching at the visual entity level has drawn attention of researchers. Nguyen *et al.* [69] developed a gaze-based system to assist note-taking on the potentially useful lecture contents (text entities) in slides. Jung *et al.* [116] presented a video processing system called Dynamic Slide that linked text entities and the corresponding speech text. Although these works can create annotations and explanations at the visual entity level, they are incompetent for the structural analysis of slides. The most important task in the structural analysis of slides is to extract and identify the types of visual entities. However, these works are only able to process text entities in the slides, leaving other types of entities being ignored. Another problem is that these works utilize hand-crafted rules to extract visual entities so that they are unsuitable for changing styles of slides. In terms of text entities, even the state-of-the-art text detection frameworks [2, 14, 94] cannot restore text entities in their hierarchical relationship, as they can only provide flat text detection. Thus, the completeness and preciseness of the knowledge points formed by the text contents cannot be guaranteed. In our approach, the hierarchical structure of knowledge points is ensured by our visual saliency clustering step.

5.2.2 Presentation Slide Processing

The most similar work of slide processing is image processing of documents. Oliveira *et al.* [117] trained a Fast R-CNN classifier for document structure analysis, which divided large chunks into text, images, and tables. Oliveira *et al.* [118] trained a CNN-based neural network 'dhSegment' for structural analysis and processing of historical documents. Shin *et al.* [119] analyzed the visual entity layout of the blackboard lectures and used the time-continuous characteristics of the blackboard to segment visual entities into formulas, texts or graphs.

Although there are several methods for document image processing [120, 121], presentation slides are very different from document images. First of all, visual entity classification based on document structure analysis pays more attention to the overall layout of the document. However, because of the irregular structure of the slides, analyzing the positional relationship and layout between the visual entities in slides becomes an indispensable step. Secondly, the entities in traditional documents are better structured and contain more rigid alignment and arrangement rules, whereas entities in slides are more staggered and unconstrained. This means that the extraction method should take more slide-based features into account when extracting visual entities.

5.2.3 Table of Contents and Note Generation

Table of contents and lecture notes are good ways to get a quick overview for previewing or browsing self-paced learning materials. A recent study [114] shows that these digital learning supports enable students to easily access and retrieve information and form an organizational personal knowledge base. There is also a rich thread of research in [38, 122] which generates tables of contents for educational videos. A table of contents helps users navigate and locate video content and enriches their way of interacting with videos. It provides a textbook-like summary catalogue for non-linear navigation and video content searches. However, the accuracy of creating such a multi-level TOCs from a slide-based educational video depends on the accuracy of visual entity recognition and the detection of hierarchical relationships.

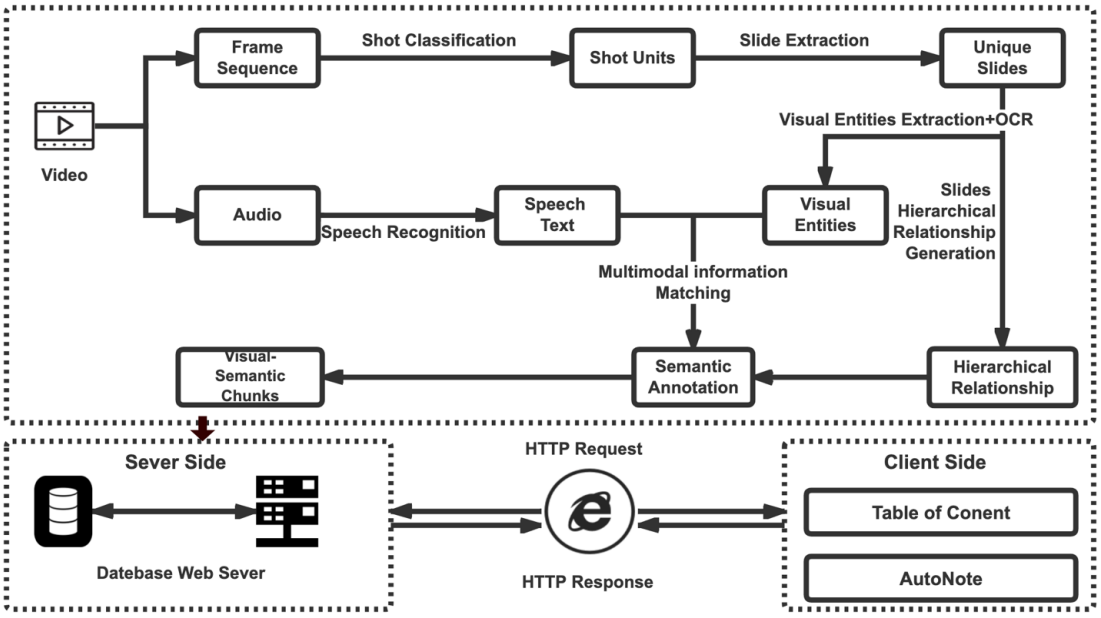


Figure 5.1 : The architecture of the proposed annotation pipeline

Shin *et al.* [119] presented a method of generating straightforward lecture notes for blackboard-style lecture videos. It established the correspondence between the visual entities and the subtitles according to the time they emerged. It is worth mentioning that the method of Shin *et al.* worked in a specific domain (blackboard style). They showed their notes on possibly infinite plain paper in a sequential format by interleaving visual entities with declarative sentences. However, learners often prefer their learning materials of limited size. Our proposed system can perform a visual entity level matching by precisely locating and analyzing their hierarchical structure, and generating the fine-grained table of contents and notes.

5.3 Visual Entity Extraction and Recognition

Figure 5.1 shows the overall architecture of the proposed annotation pipeline. Given an educational video, we process the data in both visual and audio channel. We use optical character recognition (OCR) and automatic speech recognition (ASR) technology to recognize textual data from both the visual and audio channels.

In this section, we first present our solution for visual entity extraction and

classification. This includes identifying the slide video, extracting visual entities from the slides, and classifying the types of the visual entities.

5.3.1 Slide Extraction

The emphasis of the slides in educational videos is a summary of the entire content, which can be seen as a refinement of the essence of the video. Therefore, the first step in extracting presentation slides is to know which shots contain presentation slides. In this work, shots of slides are extracted from videos by carrying out shot detection and classification using the approach described in [8]. This identifies slide shots from the whole video.

When we have identified the slide shots, we then need to pick the most significant frame in each slide shot as the slide to be extracted. In general, the contents in a slide appear sequentially. For this reason, after we obtain a shot of one slide, the last frame of that shot usually displays relatively complete contents of that shot. However, there are some counterexamples in reality. For example, the last frame is not relatively complete when the lecturer needs to reemphasize some concepts that have already been explained. In this chapter, we calculate the local maximum of the edge-based frame difference with the first frame in each shot and define the most significant frame.

The edge-based frame difference is calculated by Earth Mover's Distance (EMD) [30] between the current frame and the first frame, represented by f_1 and f_i, respectively. Both are processed by a Canny edge detector. Each frame is divided into $\mathcal{N}$ blocks, and each block contains $\mathcal{P}$ pixels. The reason of doing this is to record the extent of changes in each block. Thus, the edge-based frame difference, denoted as $\mathcal{D}_e(f_1, f_i)$, can be defined by:

$$\mathcal{D}_e(f_1, f_i) = \frac{1}{\mathcal{N} \times \mathcal{P}} \sum_{n=1}^{\mathcal{N}} \sum_{m=1}^{\mathcal{P}} d_m^{1,i}, \tag{5.1}$$

where $d_m^{1,i}$ is the m-order difference of pixel values between frame 1 and frame i in n-th block.

Let $\mathcal{D}_{max}$ denote the local maximum of the edge-based frame difference in a shot, and f_s be the most significant frame. The most significant frame f_s should let the edge-based frame difference $\mathcal{D}_e(f_1, f_s)$ reach the local maximum, *i.e.*,

$$\mathcal{D}_{max} = \max\{\ \mathcal{D}_e(f_1, f_2), ..., \mathcal{D}_e(f_1, f_n)\}\ = \mathcal{D}_e(f_1, f_s). \tag{5.2}$$

We use $\mathcal{S}$ to represent the remaining slides extracted from videos, where f_s^1 is the slide from the first shot and so on, and there are l shots in the whole video, *i.e.*,

$$\mathcal{S} = \left\{f_s^1, f_s^2, f_s^3, ..., f_s^l\right\}. \tag{5.3}$$

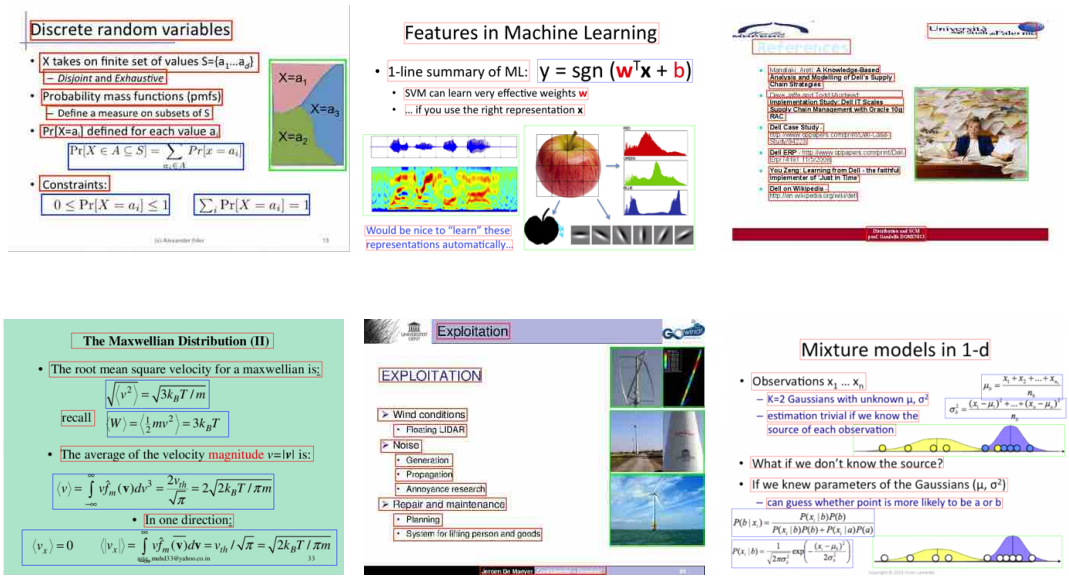


Figure 5.2 : Visualization of our visual entity extraction. The extracted text, formula and graph entities are enclosed in red, blue and green boxes, respectively.

5.3.2 Visual Entity Extraction

The visual entity extraction seems to be a general object detection problem. However, the formula and text entities cannot be perfectly retrieved through the existing object detection methods [24, 123], since the text and formula entities have their unique aspect ratios and characteristics.

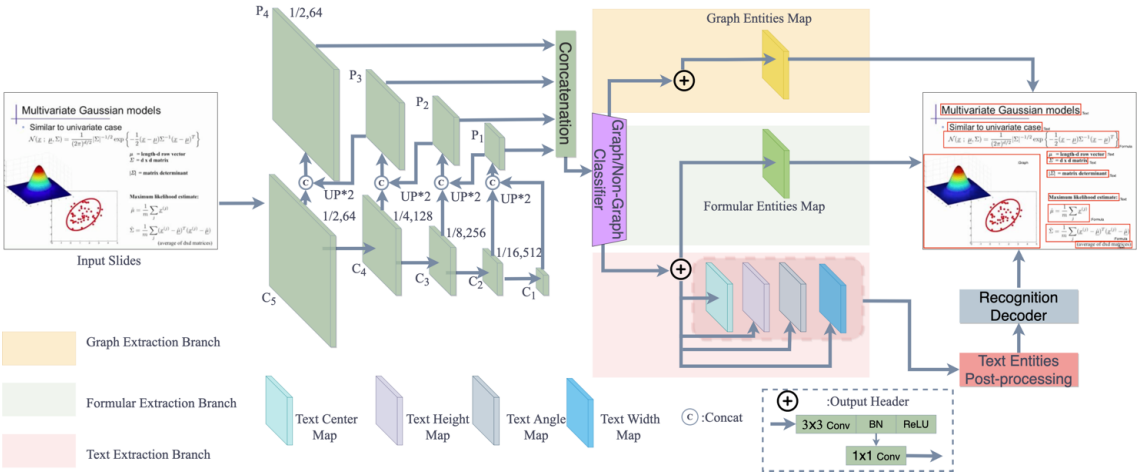


Figure 5.3 : The overall structure of our visual entity extraction network.

Targeting on this, we design our visual entity extraction framework. The overall structure of our framework is shown in Figure 5.3. The backbone of the network is ResNet50 [73] pre-trained on ImageNet. Then, we adopt the commonly used neck structure FPN [53] to capture the multi-scale features from visual entities. Different from the original FPN, here, we add an additional down-sampling layer C_1 and an up-sampling layer P_4 to the original FPN to better depict the features of visual entities, where the "$1/2, 64$" and "$1/16, 512$" indicate the scale ratios of images and the channel numbers. After that, the extracted features of $P_1 - P_4$ are concatenated and sent to the Graph/Non-Graph Classifier. The design logic of the Graph/Non-Graph Classifier is similar to the Region Proposal Network in [16, 24], and it is used to help the network to classify the graph and non-graph. This is because graph entities have unique features compared to formula and text entities.

Finally, the graph entities are extracted from the Graph Entities Map. For non-graph entities, the formula entities are extracted by analyzing the results of the Formular Entities Map. Here, in order to accurately extract text entities, we add four text maps, *i.e.*, the Text Center Line Map, Text Height Map, Text Angle Map and Text Width Map, to indicate the geometric features of the bounding box of text entities in the post-processing step. Finally, an attention-based decoder [124] is used to recognize the contents of the text entities.

The overall objective function consists of five parts:

$$L = \lambda_1 L_{Cls} + \lambda_2 L_{Graph} + \lambda_3 L_{Formula} + \lambda_4 L_{Geo}, \tag{5.4}$$

where the text geometric feature L_{geo} is calculated by:

$$L_{Geo} = L_H + L_\theta + L_W + L_{TCL}. \tag{5.5}$$

In Eq. 5.4, L_{Cls} is the classification loss to classify graph and non-graph entities, and it is a cross-entropy loss. L_{Graph}, $L_{Formula}$ and L_{TCL} are balance-class cross-entropy losses for the score maps of graph entities, formula entities and text centre lines, respectively. OHEM [77] is adopted for training L_{Graph} and L_{TCL} keeping the ratio of positive sample number and negative sample number to 1:3. λ_1, λ_2, λ_3 and λ_4 are set to 2, 1, 1 and 1, respectively. In Eq. 5.5, L_H, L_θ and L_W represent the losses of the text height, rotation angle and text width, respectively, and they are all Smoothed-L1 losses.

As shown in Figure 5.2, the visual entities are labelled as bounding boxes in the slides. The proposed deep-learning-based visual entity extraction framework is an end-to-end system, which extensively alleviates the error accumulation problem of the hand-crafted features in the traditional visual entity extraction methods. Moreover, our network computes the geometric feature of text entities to ensure that the bounding boxes press close to the real border of the text entities. Accurate bounding boxes improve the accuracy of the hierarchical structure analysis in the following steps.

5.4 Hierarchical Relationship Generation

After extracting the visual entities on the slides, we need to know their hierarchical relationship, especially for text entities and formula entities, of which hierarchical relationships are particularly important. This is because the lecture is usually expected to follow a certain hierarchical structure when explaining knowledge points so as to help students to understand the knowledge framework effectively.

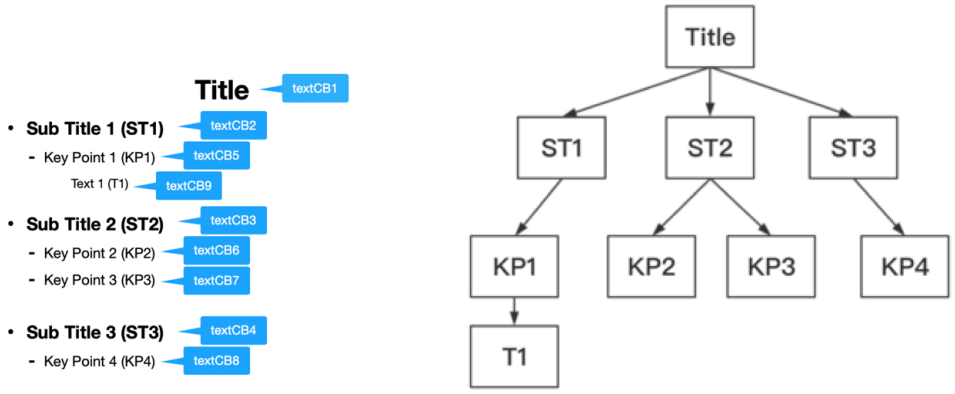
Hierarchical relationship detection in slides has broader application prospects for educational video processing, such as automatic table-of-contents generation, summarization, and note generation. However, the key step for achieving these goals is to extract the hierarchical relationships of the recognized visual entities. For example, the headline of the slide provides an important summary of the knowledge points of the lecture. Effective hierarchical relationship detection of fine granularity allows us to build more accurate and efficient information processing frameworks in educational videos.

In order to extract the hierarchical relationship after obtaining the bounding boxes of all visual entities, we define some features of the bounding boxes and the visual entities. These are shown in Table 5.1.

Table 5.1 : Features used to extract the hierarchical relationship

Name	Type	Meaning
id	int	Sequence number of the visual entity
$type$	string	Text — Formula — Graph
$left$	float	Bounding box' left position
top	float	Bounding box' top position
$width$	float	Width of the bounding box
$height$	int	Height of the bounding box
$text$	string	Text in the bounding box

The layout of the slides is designed by the lecturer and can reflect the hierarchical relationship between the text content blocks (hereinafter referred to as $textCB$). The higher level in a hierarchical relationship, the greater the refinement of the knowledge points is. Here, we use visual saliency as the representation of the level of hierarchical relationship. The top level of hierarchical relationship has the largest visual saliency. Figure 5.4(a) shows a typical slide's layout. In the figure, $textCB^1$ is the headline of the slide, $textCB^{2-4}$ are the second-level titles, $textCB^{5-8}$ are the knowledge points under the second-level titles, and $textCB^9$ is the descriptive text below the third level. Based on these relationships, a hierarchical relationship tree can be generated, as shown in Figure 5.4(b).



(a) The layout of a slide

(b) The extracted hierarchical relationship of (a)

Figure 5.4 : An example of the typical layout of a slide (a) and the hierarchical relationship extracted between its text content blocks (b).

5.4.1 Visual Saliency of Visual Entities

In this work, we mainly consider the hierarchical relationship between text entities because they play the key role in understanding the structure of the lecture. Although the bounding boxes of text entities contain features of the text contents, their positions and sizes cannot represent all features of text entity comprehensively. Hence, we define three more specific features to measure text entities.

Font size $\mathcal{T}$: Font size is a very important feature for understanding the layout of a slide because text with a larger font usually indicates that it is the highlight of the slide presentation. The size of a font can be determined by the height of the characters. According to the rules of writing English letters, letters can be divided into three categories based on the their heights, as shown in Figure 5.5. An intuitive way to calculate the size of a font is to calculate the average height of all letters. However, we find that this way is very inaccurate because the proportion of different types of letters in a text block would have a big impact on the resultant font size value. In this chapter, we propose to only calculate the height of Class 1 letters (see Figure 5.5) because almost all words contain letters from Class 1.

Indentation $\mathcal{I}$: Lecturers often use text indentation to show the hierarchical rela-

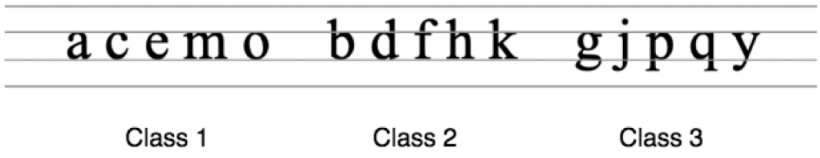


Figure 5.5 : The three classes of English letters

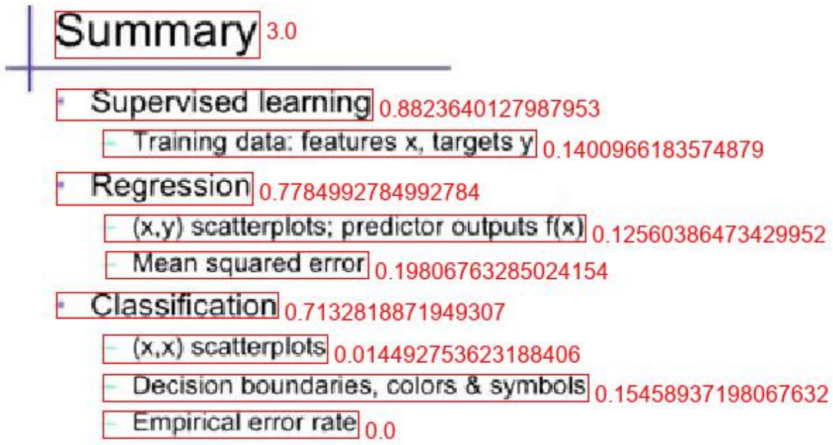


Figure 5.6 : Visual saliency scores for visual entities on a slide

tionship of contents. Indentation refers to the distance between the leftmost side of the text and the page boundary. In general, the larger the indentation is, the less important the text is. Here, we use the left position of the bounding box of the text entity $textCB.left$ as the indentation.

Boldface fonts $\mathcal{B}$: Bold fonts are often used to emphasize important contents. The overall width of the bold font does not necessarily change, but the width of the stroke will increase significantly. If the text entity is binarized, widening the stroke means that the number of black pixels increases. Therefore, the average number of black pixels per character can be used as a representative of the thickness feature, which can be calculated using the number of black pixels in the text entities divided by the number of characters in text entities.

Hence, let $\mathcal{T}'$, $\mathcal{I}'$ and $\mathcal{B}'$ denote the results after min-max normalization on $\mathcal{T}$,

$\mathcal{I}$ and $\mathcal{B}$, respectively. The visual saliency and its score for a visual entity can be defined by:

$$textCB.saliency = [\mathcal{T}', \mathcal{B}', \mathcal{I}'], \tag{5.6}$$

$$textCB.score = \mathcal{T}' + \mathcal{B}' + \frac{1}{\mathcal{I}'}. \tag{5.7}$$

The visual saliency scores of an exemplar text slide are shown in Figure 5.6.

5.4.2 Visual Saliency Clustering

Text entities belonging to the same hierarchy usually have similar visual saliency, whereas text blocks at different levels tend to have large differences in their font size, indentation, and thickness of the text. In this work, we propose to adopt the idea of clustering to classify different hierarchies of text entities into their corresponding hierarchical classes. Since the numbers of hierarchies on different slides are not fixed or predictable, the clustering algorithm needs to work with an unspecified number of clusters. In this chapter, we adopt the Affinity Propagation (AP) clustering algorithm [31], which is based on "message passing" between data points. The AP algorithm does not need to have any prior knowledge about the number of clusters. So, it is appropriate for clustering the hierarchy of text entities on slides. Next, we present the details of how we formulate our problem in the AP algorithm.

Let $textCB.saliency$ denote the visual saliency vector of the text entity $textCB$, and $\mathcal{S}$ be a similarity matrix of the text entities on the slide. The similarity between two text entities $textCB^n$ and $textCB^m$, denoted as $\mathcal{S}[n, m]$, can be defined by:

$$\mathcal{S}[n, m] = - \left\| textCB^n.saliency - textCB^m.saliency \right\|^2. \tag{5.8}$$

Denote the responsibility matrix and availability matrix in the AP algorithm as $\mathcal{R}$ and $\mathcal{A}$, respectively. The update rules can be written as:

$$\mathcal{R}[n, m] \leftarrow \mathcal{S}[n, m] - \max_{m' \neq m} (\mathcal{A}[n, m'] + \mathcal{S}[n, m']) \tag{5.9}$$

and

$$\mathcal{A}(n,m) \leftarrow \begin{cases} \sum\limits_{n' \neq m, n=m} \max(0, \mathcal{R}[n', m]) \\ \min\limits_{n \neq m}(0, \mathcal{R}[m,m] + \sum\limits_{n' \notin \{n,m\}} \max(0, \mathcal{R}[n', m])). \end{cases} \tag{5.10}$$

5.4.3 Dependency Relation Detection

Having clustered the text entities on the slide, we have not yet established a dependency relationship between each cluster. Therefore, the main task of this subsection is to rank the clusters as well as to look for the potential 'parent' of each cluster.

First, for a set of clusters $\mathcal{C} = [c_1, c_2, c_3 ... c_n]$, in order to rank the clusters in the set $\mathcal{C}$, we compute the average visual saliency score of each hierarchical cluster c_i. The rank of $\mathcal{C}$ reveals the hierarchy of clusters, meaning that the hierarchy of all of the text entities is found.

Then, the next step is to establish a relationship between different levels of the hierarchy, *i.e.*, to find the 'parent' and 'child' entities for each visual entity on the slide. For the visual entity in the l-th level, its parent level is at $(l-1)$-th level. This is because the layout of the slideshow usually shows its 'child' contents immediately after the previous level of the content. We can use the vertical position to calculate the distance of the text content blocks between adjacent levels.

Let *dist* be the vertical distance between two visual entities. For visual entity $textCB_l^i$ on the l-th level and visual entity $textCB_{l-1}^j$ on the $(l-1)$-th level, the vertical distance is defined by:

$$dist(textCB_l^i, textCB_{l-1}^j) = |textCB_l^i.top - textCB_{l-1}^j.top|. \tag{5.11}$$

The algorithm for detecting the dependency relationship is shown in Algorithm 3.

Algorithm 3: Dependency Relationship Detection

Input: Set of clusters $\mathcal{C} = [c_1, c_2...c_l...c_n]$; Headline $rootCB$
Output: List of visual entities with the parent entity
$rootCB.parent \leftarrow null$
for *each* $c_l \in \mathcal{C}$ **do**
  Sort the visual entities in c_l based on their top positions
  for *each* $textCB^i \in c_l$ **do**
    if $l == 1$ **then**
      $textCB^i.parent \leftarrow rootCB$
    else
      $S = list()$
      for *each* $textCB^j \in c_{l-1}$ **do**
        if $textCB^j.top < textCB^i.top$ **then**
          $S.add(textCB^j)$
        end
      end
      $dist_{min} = -\infty$
      for *each* $textCB^k \in S$ **do**
        if $dist(textCB^k, textCB^i) > dist_{min}$ **then**
          $dist_{min} = dist(textCB^k, textCB^i)$
          $textCB^{parent} = textCB^k$
        end
      end
      $textCB^i.parent \leftarrow textCB^{parent}$
    end
  end
end

5.5 Semantic Annotation of Visual Entities based on Multi-Channel Information

In educational videos, the audio channel information plays a critical role in video annotation. In most cases, the lecturer explains every visual entity on the slide, meaning that it is the speech text information in the audio channel that helps learners to understand the key contents. However, with the existing time-alignment based solutions, the correspondence between a single visual entity and its speech text information can only be annotated by aligning the time when these visual entities appear on the slides one by one. To achieve accurate visual entity annotation, in this chapter we propose to calculate the semantic similarity between each visual entity

and the interpretation content based on the result of time-alignment.

Currently, Word Vectors [125, 126] have been widely used to indicate semantic information from textual data. They used pre-trained word vectors that were trained on Wikipedia using fastText [127] to maximize the preservation of semantic information in lecture videos. Since the contents in the audio channel are sentences, in order to calculate the semantic similarity between visual entities and their explanation, in this chapter we adopt the Word Mover's Distance (WMD) [128] to calculate the similarity between the entities and sentences.

WMD is an algorithm used to calculate the similarity between document vectors based on word vectors. Its semantic representation can be based on the word vector obtained by fastText. The lecturer's explanation is usually considered as an extension of the contents of the slide. The process of the lecturer's explanation can be seen as that the contents on the slides are 'moved' and expanded on the basis of the original contents. This is analogous to the principle of the WMD in the way that, the distance from the words of a paragraph 'travels' to the words of another paragraph. A shorter WMD distance indicates higher similarity of paragraphs.

As the semantic information of the graph entity is hard to capture, here we first consider creating annotations for text and formula entities with graph entities being processed subsequent to the result for the first two types of entities. We choose to merge some of the text entities, based on their hierarchical relationships detected in the previous step, to preserve the integrity of the knowledge points to the greatest extent. This means the text entities below the subtitle of hierarchical relationships will be merged.

In the typical hierarchical layout in Figure 5.4, KP1 and T1 will be merged into ST1. Because the slides are usually presented from top to bottom and from left to right, we number the visual entities in this order. Therefore, the first visual entity corresponds to speech text that appears first, and the second visual entity corresponds to the following parts of the remaining speech texts, and so on. The best matching result can be found by maximizing the sum of similarity (minimize the total WMD distance) of all blocks into which subtitles are partitioned. Note

that the number of blocks in this method is equal to the number of visual entities, based on the assumption that one entity must correspond to at least one sentence in speech texts and that each sentence can only be used once. Figure 5.7 shows two different partition situations, where, in each case, the total WMD distance of the matching method is calculated differently.

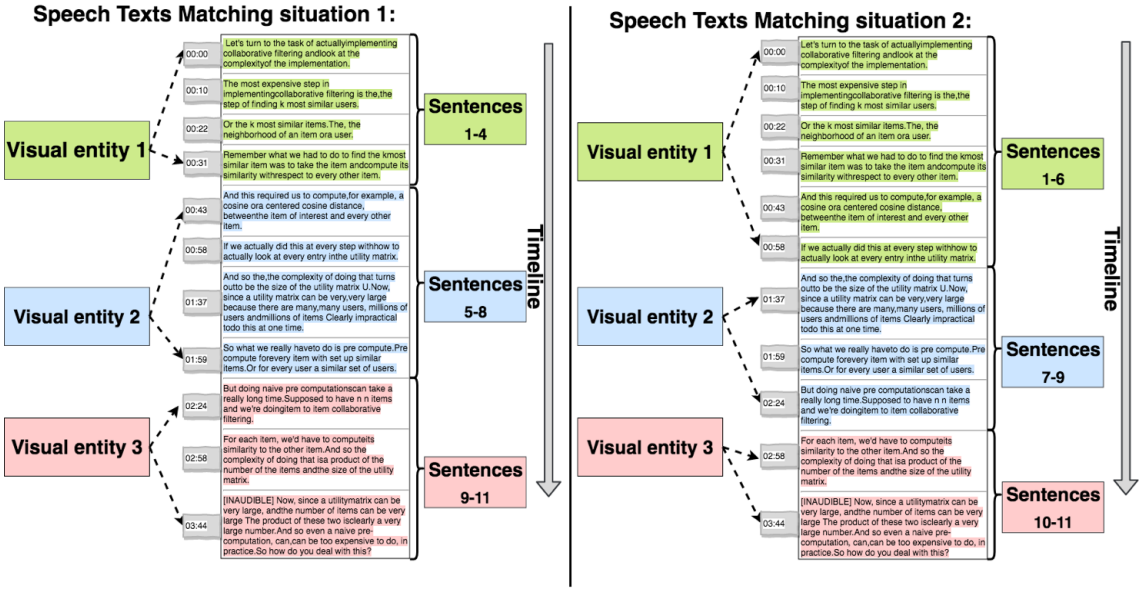


Figure 5.7 : Two different matching situations of corresponding speech texts

We use n to represent the number of visual entities, $s[1, 2, 3...t]$ to represent the sentence sets of the corresponding speech text, $\mathcal{D}_{opt}$ to represent the minimum total WMD distance that the partition approach reaches and $\mathcal{WMD}$ to represent the WMD distance. The optimal partition to minimize the total WMD distance is defined by:

$$\begin{aligned}\mathcal{D}_{opt}(n, s[1, 2, 3...t]) = min_{k<t}[(\mathcal{D}_{opt}(n-1, s[1...k]) \\ +\mathcal{WMD}(n, s[k+1...t])],\end{aligned} \tag{5.12}$$

where $\mathcal{D}_{opt}(n, s[1, 2, 3...t])$ means the total minimal WMD distance when we need to divide the first t sentences in the speech text set s into n blocks for n visual entities; $\mathcal{D}_{opt}(n-1, s[1...k])$ means the total minimal WMD distance when we need to divide the first k sentences in the sentence set s into $n-1$ blocks; and $\mathcal{WMD}(n, s[k+1, ...t])$

means the WMD distance of sentences from $s[k+1, ...t]$ to the n-th entity.

To evaluate the WMD distance between the formula entities and their corresponding speech text, in this chapter we first convert formula entities into natural language in the order of left to right and top to bottom. An example is: $y = sign(w^T x + b) \rightarrow$ 'y equal sign w transpose x plus b'.

For slides with graph entities, since we cannot extract perfectly the semantic information of a picture, it is challenging to calculate the semantic similarity between the images and their related explanatory contents. The most accurate annotation of the graph is created by time-alignment methods according to the appearance of these entities. For circumstances where time alignment fails, there are sometimes obvious suggestive words describing images as well as the location information of this graph such as image, graph, figures and so on. Therefore, in our work, we first extract only the text and formula entities in the slides and use them to pair speech texts for the entire slide. Then, we merge the graph entities with one of the text or formula entities, based on their suggestive words, or their synonyms, of this image entity.

A visualized example is shown in Figure 5.8. The Graph entity 1 is merged with the Non-graph entity 1 because the corresponding speech text contains the suggestive word 'graph'.

5.6 Annotation Tools and Applications based on Visual Entities for Educational Video

5.6.1 AutoNote Generation

To generate properly-structured and key-points highlighted lecture notes, our layout method partitions a plain note page into blocks according to the importance of visual entities. To ensure the readability of the notes after layout partitioning, the size of the block for each visual entity must satisfy a certain aspect ratio. Meanwhile the page area should be used as completely as possible. This is achieved by carefully designing the area ratio of the input and adding reshaping function in the process

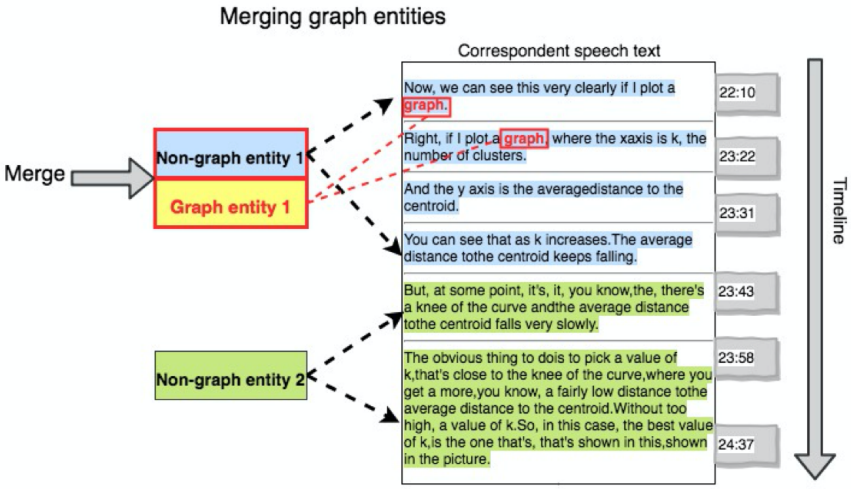


Figure 5.8 : A visual example of merging graph entities

of original Squarified Treemaps algorithm [129]. This Treemaps algorithm takes the number of blocks that need to be divided and their area information as the input in order to generate approximate-to-square and flexible-size blocks.

Designing the area ratio of the input

In general, critical visual entities are those that the lecturer spends more time in explaining. This can be measured by the length of the explaining contents. At the same time, the size of the visual entity and whether it contains formulas or pictures are also important factors in the weight of the area of blocks. Based on these, we design following weight measurement function as:

$$\mathcal{I}(\mathcal{V}) = \alpha(\frac{w_v \times h_v}{\mathcal{W}_{slide} \times \mathcal{H}_{slide}}) + \beta\mathcal{R}_e(\mathcal{V}) + (1-\gamma)\mathcal{R}_v(\mathcal{V}) + \gamma\mathcal{C}, \tag{5.13}$$

where w_v and h_v are the weight and height of entity $\mathcal{V}$; $\mathcal{W}_{slide}$ and $\mathcal{H}_{slide}$ are weight and height of the slide; $\mathcal{R}_e(\mathcal{V})$ is the ratio of the explanatory speech texts of $\mathcal{V}$ to all explanatory speech texts in the slide ; and $\mathcal{R}_v(\mathcal{V})$ is the ratio of textual contents of $\mathcal{V}$ to all textual contents in the slide. if $\mathcal{V}$ is formula or graph entity, γ equal to 1, else 0. $\mathcal{C}$ is an additional constant for graph and formula entities. In the experiment of this chapter, α, β and $\mathcal{C}$ are set at 0.53, 0.27 and 0.2 respectively.

Algorithm 4: Generate note-like blocks

Function Layout $(\mathcal{L}, x1, y1, x2, y2)$;
Input: List of weights for blocks in note:$\mathcal{L}$, Diagonal coordinates of note area:$(x1, y1), (x2, y2)$
Output: note layout
$\mathcal{L} = \frac{\mathcal{L}}{\mathcal{L}.sum()}$; `// weight normalization`
if $\mathcal{L}.lenth = 0$ **then**
  return
end
if $\mathcal{L}.lengh = 1$ **then**
  $Draw(\mathcal{L}, x1, y1, x2, y2)$; `// If there is only one block in note`
end
$i = 1$;
while $Worst(\mathcal{L}[: i], x1, y1, x2, y2) >= Worst(\mathcal{L}[: i+1], x1, y1, x2, y2)$ **do**
  $i = i + 1$; `// the` $\mathcal{L}[i+1]$ `should be added to the previous` $\mathcal{L}[i]$ `if it has less aspect ratio`
end
if $\mathcal{L}[: i].lengh > 1$ **then**
  $DrawReshape(\mathcal{L}[: i], x1, y1, x2, y2)$; `// rectangle to trapezoid`
end
$x1^*, y1^*, x2^*, y2^* = Left(\mathcal{L}[i :], x1, y1, x2, y2)$; `// calculate remaining area on note`
layout($\mathcal{L}[i :], x1^*, y1^*, x2^*, y2^*$**);**

We design $Layout(\mathcal{L}, x1, y1, x2, y2)$, a recursive function that can generate note-like layout (Algorithm 4). $\mathcal{L}$ is the list of weights for blocks, which can be calculated by the weight measurement function. $\mathcal{L}[: i]$ means getting the first i elements from list $\mathcal{L}$. $x1, y1, x2, y2$ are diagonal coordinates of the initial note area. $Draw(\mathcal{L}, x1, y1, x2, y2)$ means drawing rectangles in the note area determined by $x1, y1, x2, y2$. $Worst(\mathcal{L}[: i], x1, y1, x2, y2)$ is a function meaning the worst aspect ratio of i-th block in $\mathcal{L}$. This function keeping $x1, y1, x2, y2$ is to get the weigh or hight information to calculate worst aspect ratio. The detail of how to calculate worst aspect ratio can be found in the paper of Bruls et al. [129]. When $Worst(\mathcal{L}[: i], x1, y1, x2, y2) >= Worst(\mathcal{L}[: i+1], x1, y1, x2, y2)$, the newly generated rectangle is added to the column of the previously generated rectangle. The function $Left(\mathcal{L}[i :], x1, y1, x2, y2)$ means calculating the diagonal coordinates for remaining note area if we have already finished drawing the first i blocks in the

note area. The $DrawReshape(\mathcal{L}[:i], x1, y1, x2, y2)$ will reshape the rectangles into trapezoid with same area.

Filling the visual entities into blocks

To ensure the readability and usability of notes, we match one-page note to one slide by default. If there are few visual entities in a slide (less than three), this slide will be placed in the same page together with the visual entities on the following slide, ensuring that the number of visual entities in each note does not exceed eight and font size is no less than ten. The visual entity and explanation are aligned and packed into the block from top to bottom or from left to right according to the remaining area in the block. Moreover, the interval size of element should be as large as possible. The important key words are highlighted in red by calculating the weight $w(t)$ for each word t. $tfidf$ stands for term frequency-inverse document frequency. Let $\mathcal{V}^*$ and $\mathcal{T}^*$ be the corpus of text from visual(slides) and audio(subtitle) channels, and $\mathcal{V}$ and $\mathcal{T}$ be the corpus of text of each slide and its speech texts. The visual texts in slide are generally more important than those in audio channel, leading to that the γ is set to 0.7. The formula used for calculating the weight $w(t)$ is:

$$w(t) = \gamma \frac{tfidf(t, \mathcal{V}, \mathcal{V}^*)}{\sum_{t_i \in V^*} tfidf(t_i, \mathcal{V}, \mathcal{V}^*)} + (1-\gamma) \frac{tfidf(t, \mathcal{T}, \mathcal{T}^*)}{\sum_{t_j \in T^*} tfidf(t_j, \mathcal{T}, \mathcal{T}^*)}. \tag{5.14}$$

5.6.2 Table-of-Contents Generation

With the visual entities being recognized and their hierarchical relationship being detected, it is not hard to generate a detailed ToC with all of the knowledge points for an educational video. A Table of Contents (ToC) contains important features of documents for summarization and navigation. Generating a multi-level textbook-like ToC for educational videos can help students understand the video contents of the course more quickly and facilitate efficient navigation to specific areas of the knowledge points.

An example is shown in Figure 5.9(a), where input slides are processed using our proposed approach. With the extracted hierarchical relationship among visual entities, a TOC of the input slides is generated to the right.

5.7 Experiments

To evaluate the effectiveness of our proposed approach, in this section, experiments are conducted on public benchmark datasets. In this section, we first introduce the datasets tested in the experiments and the implementation details. Then, quantitative comparisons with the existing approaches and user study are presented. Finally, we further discuss why our proposed methods are effective for locating information and its limitation.

5.7.1 Datasets

ICDAR2013: The ICDAR2013 dataset [130] consists of text images of digital documents (Web and Email), with 229 training and 223 testing images.

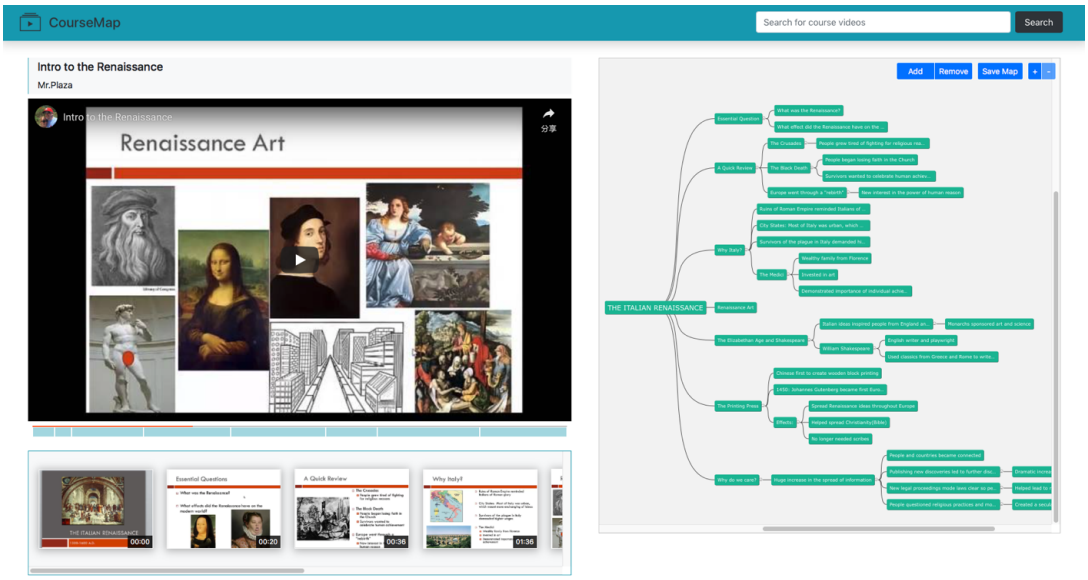
ISI-PPT: The ISI-PPT dataset [131], consisting of 10,692 images, was created for detecting Arabic and English texts in presentation slides.

SPaSe: The SPaSe dataset [132], consisting of 1,500 training images and 500 testing images, was created for slide layout semantic segmentation.

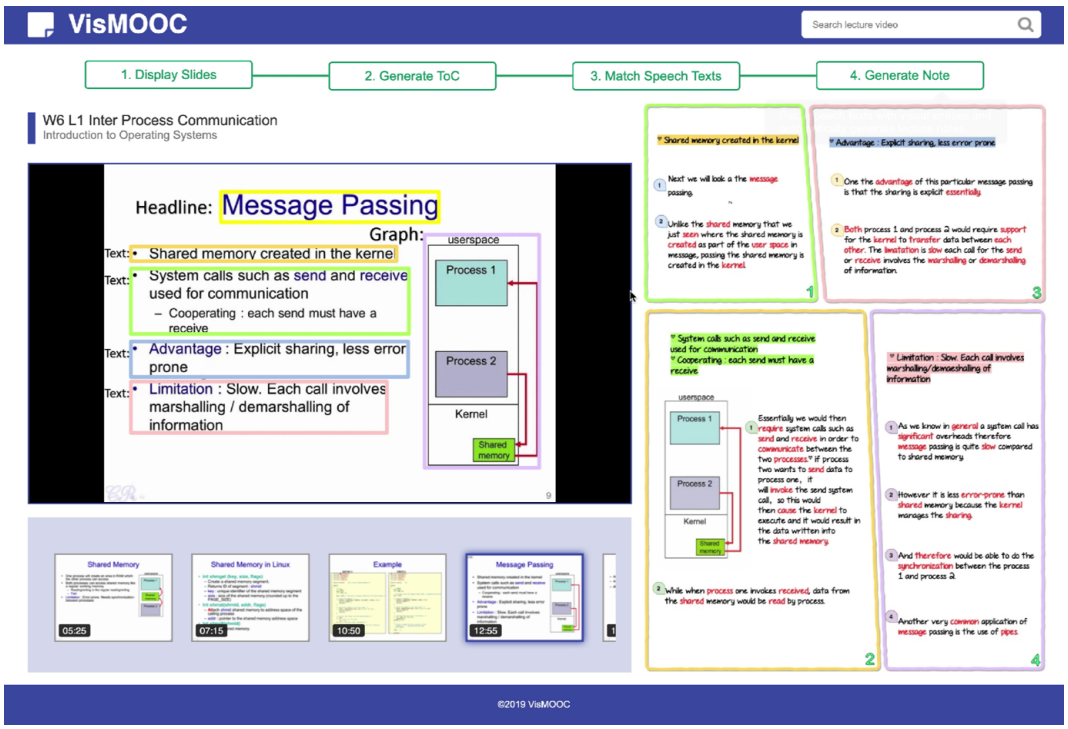
Slideshare-1M: The Slideshare-1M [133] is a dataset that is widely used for image retrieval, and it consists of 977,605 raw slide images related to engineering and science.

5.7.2 Implementation Details

Our visual entity extraction model is implemented using PyTorch 1.7. The backbone of our model is ResNet50 [73] pre-trained on ImageNet. We conduct our experiments on NVIDIA RTX 3090 GPU with 24GB memory. The common data augmentation techniques are adopted, including rotation, cropping, colour variations, *etc*. Since text entities play a more important role in the slides for the constructing of knowledge points, here, we use SynthText and ICDAR2017-MLT to pre-train the text and formula classification branch in our model to enrich the features of



(a) Table-of-Contents Generation



(b) Automatic Note Creation

Figure 5.9 : Examples of the proposed navigation tool and its applications based on visual entities for educational videos

text instances for more accurate text entity extraction. We first train our model for ten epochs on SynthText and then 50 epochs on ICDAR2017-MLT with Adam

Table 5.2 : Results of visual entity extraction on ISI-PPT. (*: multi-scale training or testing.)

Method	P(%)	R(%)	F(%)
Text-Block FCN [134]	63.0	67.0	64.9
Xu *et al.* [135]	57.8	62.7	60.1
Wu *et al.* [131]	91.0	94.0	92.5
*Wu *et al.* [131]	94.0	**97.0**	95.5
*Ours	**96.5**	96.8	**96.6**

optimizer at a learning rate of 0.001. Then, we fine-tune our model on our manually collected dataset. We first collect 819 slides from computer science lectures since they have a good inclusion of an appropriate mix of graph, formula and text entities. Here, the slides are selected from the course videos on YouTube, and for each video, we select no more than five slides to ensure diversity. Moreover, we follow the method in [132] and use Slideshare-1M to further extend our dataset. We select 501 slides from different presentations in Slideshare-1M and add them to our manually collected dataset. We simply split 1,000 slides into the training set and 320 into the testing set. The ground truth for each entity in slides is created manually. The text entities are labelled at a single line level. For training, all of the slides are resized to 640×640 pixels. For testing, we evaluate the performance of our visual entity extraction model on ISI-PPT, ICDAR2013, and our manually collected dataset, respectively.

5.7.3 Evaluation of Visual Entity Extraction

We first evaluate our visual entity extraction model on the ISI-PPT dataset. Here, we follow the dataset split rule in Wu [131] and use 90% of the images for training. The results shown in Table 5.2 show that our method outperformed the existing state-of-the-art methods with an F-measure of 96.6%, showing that our method can accurately extract the text entities in the slide images.

We also evaluate our visual entity extraction model on the ICDAR2013 dataset as the web and e-mail pages in this dataset are similar to the slides. The detection results are shown in Table 5.3. Compared with state-of-the-art text detection

Table 5.3 : Results of visual entity extraction on ICDAR2013. (*: multi-scale training or testing)

Method	Backbone	P(%)	R(%)	F(%)
CTPN [136]	VGG16	93.0	83.0	88.0
SegLink [22]	VGG16	87.7	83.0	85.3
PixelLink [137]	VGG16	88.6	87.5	88.1
ATRR [12]	VGG16	93.7	89.7	91.7
*SPCNet [138]	ResNet50	93.8	**90.5**	92.1
*TextFuseNet [2]	ResNet50	**95.1**	89.5	**92.2**
*Ours	ResNet50	94.4	89.7	92.0

methods, our method has also achieved a comparable SOTA result of an F-measure of 92.0%. Note that the TextFuseNet [2] needs additional char-level annotation on ICDAR2013 to train their weakly supervised char detection part, while our model only uses the word-level ground truth provided by ICDAR2013. We infer that the char-level annotation will bring additional semantic information of text, and this is why TextFuseNet performs slightly better than our method.

Then, we evaluate the effect of our proposed visual entity extraction on our manually collected dataset. The results are shown in Table 5.4. As the table shows, our deep-learning-based visual entity extraction has achieved a higher F-measure compared with the rule-based method in [135]. The experimental results also show that our proposed method is effective in text, formula and graph entity extraction. The F-measure of 96.1% for text entities also largely prevents the missing detection problem, making the hierarchical relationship extraction in the next step more accurate.

Table 5.4 : Comparison of results obtained with different approaches for visual entity extraction.

Method	Entity Type	P(%)	R(%)	F(%)
Xu *et al.* [135]	Formula	81.6	81.8	81.7
	Graph	79.5	78.7	79.1
	Text	89.5	87.3	88.3
Ours	Formula	90.1	92.3	91.2
	Graph	92.2	91.5	91.8
	Text	96.8	95.5	96.1

Table 5.5 : Performance comparison of hierarchical relationship generation using different features

Method	P(%)	R(%)	F(%)
Font size only	52.9	75.4	62.2
Indentation only	73.4	72.2	72.8
Boldface only	69.0	76.3	72.5
Ours	84.3	87.4	85.8

5.7.4 Evaluation of Hierarchical Relationship Extraction

In order to evaluate the effectiveness of the hierarchical relationship generation for slides, we select 70 slides and manually record 532 pairs of hierarchical relationships. The parent entity of each text entity is labelled manually. We evaluate the effect of using font size, indentation and boldface as the features for hierarchical relationship generation separately. The experimental results are shown in Table 5.5.

From the table, it can be seen that our method using all three types of features has achieved significantly improved accuracy of hierarchical relationship generation with an F-measure of 85.8%. The indentation feature has an F-measure of 72.8%, while the performance of the font size feature is the worst. This is because the font size of the subtitles and the body text can be very similar in some slides.

Then, we compare the performance of our method with the existing methods on the extraction of slide headlines on the SPaSe dataset, which provides the semantic segmentation ground truth of the headline of the slide. Note that SPaSe does not provide the text annotation results at a single line level, so this dataset cannot be used to evaluate our hierarchical relationship extraction method. Here, we simply adopt the minimum enclosing rectangle of these ground truths as the bounding box of the headline. We first use our visual entity extraction model to detect the possible text entities in slides and then use the hierarchical relationship extraction method to extract the headline. Thus, the text entity that has the highest visual saliency score is defined as the headline. The experimental results are shown in Table 5.6.

The table shows that our headline extraction method has achieved an F-measure of 89.8%, higher than Zhao *et al.* [2] and Yang *et al.* [18]. This is because both of

Table 5.6 : Performance comparison on headline extraction

Method	P(%)	R(%)	F(%)
Yang *et al.* [18]	77.4	82.2	79.7
Zhao *et al.* [2]	82.5	85.1	83.7
Ours	89.5	90.2	89.8

Table 5.7 : Performance comparison on matching corresponding speech text

Method	P(%)	R(%)	F(%)
Time-alignment	79.5	45.4	57.7
Ours	86.0	91.7	88.7

their methods used a hand-crafted rule-based method for locating the position of the headline, so it is unstable and easy to accumulate errors, while our method uses a deep learning model so that the bounding boxes are much more accurate to reflect the position as well as the height of text entities. Moreover, it should be noticed that Zhao *et al.* [2] assumed that the title must be the first visual entity. However, this assumption is valid only for extracting titles. For non-title contents, different slides may have great differences in the layouts and positions of visual entities.

5.7.5 Evaluation of Visual Entity Matching

For visual entity matching, we compare our method with the basic time alignment for matching visual entities and their corresponding explanatory speech texts. We select 50 slide-based educational videos, while the ground truth of the matching results and the visual entity extraction are all done manually. We use time-alignment as the baseline.

From the results shown in Table 5.7, we find that the recall of the time-alignment method is low because the time when the visual entity appears cannot be recorded in most of the cases in the test dataset. Our method has achieved higher precision and can deal with the situation when time-alignment may fail, contributing to the improved recall rate.

5.7.6 User Study

We designed a survey to assess the effectiveness of the ToCs and the notes ('AutoNote') automatically generated with our approach in facilitating learning. There are following three main tasks in our study, some of which are common evaluation methods for online lecture learning tools [38, 71, 119, 135].

Searching task: Learners need to find several topics (both textual and visual concepts that are included) in the lecture.
Detail understanding task: Learners need to answer some questions about the details in the video. This requires that the learners have a certain understanding of the content.
Summarization task: Learners need to summarize the key points in the video. This is encouraged to be written in detail. The quality of the summary created by a learner is judged by the number of pre-defined key contents covered.
In our design, the tasks were performed in the order of: 1. Searching task, 2. Detail understanding task, and 3. Summarization task.

Participants were 25 students from a local university between the ages of 20 and 30 who have online learning experience. They were divided into 5 groups and different videos used for each interface. The order that the participants performed each interface was determined by a Latin square. We provided a warm-up session for them to get familiar with the above learning interfaces. During the user study, we adopted a within-participant design. We ensured that each participant performed tasks on each interface. Each lecture video was about 7-8 minutes long. Each participant had 5 minutes to skim through the video before the experiment started, and there was a 20-minute limit for all tasks. The participants needed to record the start time and the end time of each task.

We compare the time consumed for completing the two tasks, *i.e.*, the Searching task and the Detail Understanding task using our generated ToCs and AutoNotes, with the peer interfaces of YouTube (baseline), the video with the original presentation slides and the system of Zhao *et al.* [8]. The comparison result is shown in Figure 5.10. Note that, the system of Zhao *et al.* is a navigation system that

uses information from multiple channels and presents users interactive visualization components such as table of contents and the word cloud.

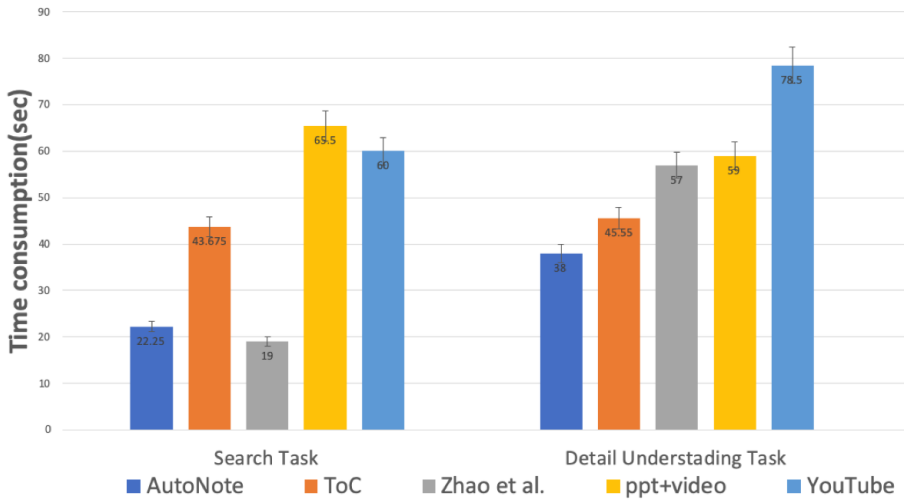


Figure 5.10 : Time consumption comparison of the Searching task and the Detail Understanding task.

As shown in this figure, for the Searching task, using both of the ToCs and AutoNotes generated with our approach, users can accomplish the task faster than using YouTube and the original slides. In contrast, using Zhao's system, users take the shortest time (19s), because this system uses a searching index for searching textual concepts.

For the Detail Understanding task, the auto-generated note has achieved the shortest average searching time (34s), followed by the ToC, which takes 48.2s on average to search the detailed concepts. Compared with YouTube (78.5s), Zhao's system (57s) and the original slide with video (59s), our method takes the hierarchical relationship of knowledge points into consideration as well as creating annotations for visual entities in slides. The differences across the interface test for the searching task and the detail understanding task are significant with ANOVA test results ($p < .0001$). In the Searching task, the Post-Hoc Test reveals that learners take less time to find the topics using our auto-generated notes than using the original slides with videos and YouTube ($p < .0001$). In the Detail Understanding task, the Post-Hoc Test reveals that users take less time to understand the contents using our

automatically generated notes and ToCs than using other interfaces ($p < .0001$).

Figure 5.11 shows the comparison of the Summarization task. There are two types of summary, *i.e.*, main points and detailed points. The detailed points are generally some explanations that complement the main points. These two summary tasks were evaluated by the ROUGE-1 [139], which was calculated as the number of words in the summary created by the learner divided by the total words in the reference key content. The order of tasks is listed as follows. For the main points covered, according to the ANOVA test result ($p = 0.747$), our ToC and auto-generated note do not have a significant difference. However, there is a significant difference for the detailed points covered ($p < .0001$). The Post-Hoc Test reveals that the percentage of the detailed points covered using our auto-generated notes is significantly larger than using Zhao's system ($p < .001$). According to the participants' feedback, ToCs and auto-generated notes are very helpful for looking for the main points. This is because both ToCs and auto-generated notes provide a framework of knowledge points based on their hierarchical relationships. Besides, the auto-generated notes add lecturer's speech text as an annotation and supplementary explanation to the hierarchical framework. Moreover, creating ToC knowledge structure and notes is a common and popular way of learning. Our approach simplifies the creation process to a certain extent, and learners can also add personalized content.

From the above user survey, it can be concluded that both ToCs and AutoNotes can be used as supplementary material in facilitating learning. This can improve the learning efficiency.

5.7.7 Discussion of Effectiveness in Locating Information

The experiment results and user study show that our proposed system can effectively help users to locate information. Interpreting the results, we owe this to two reasons. First, we propose a deep learning based end-to-end visual entity extraction method, which eliminates the error propagation in the traditional hand-crafted methods. This method aims at accurately extracting the visual entities according to their feature differences, especially for the text entities. The SOTA-comparable

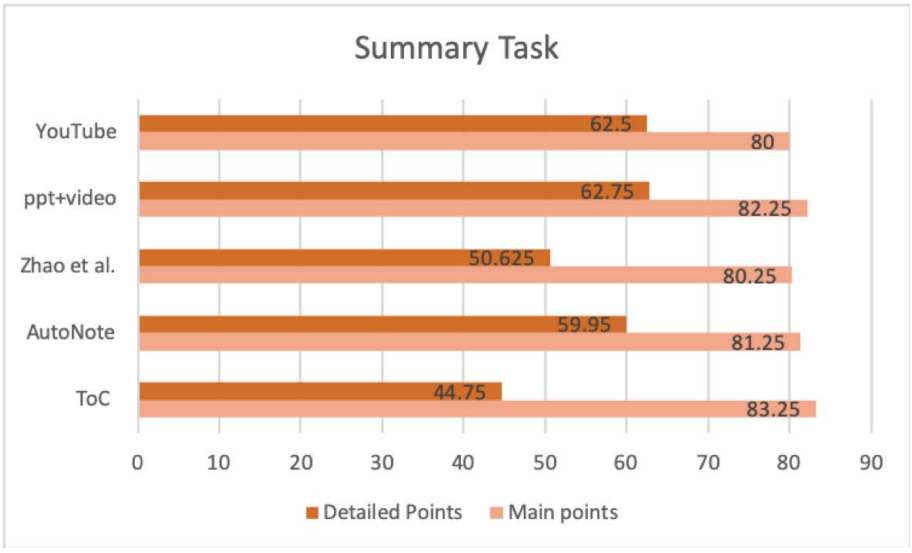


Figure 5.11 : Results of the Summarization Task

results on public benchmark datasets ICDAR2013 and ISI-PPT demonstrate this. Secondly, based on the accurate visual entity extraction results, our methods analyze the hierarchical structure of visual entities to keep the logical and structure of knowledge points complete before they are linked with the corresponding explanation.

5.7.8 Limitation

It is also worth noting that the visual entity extraction results and subtitle quality from the original video can impact the quality of the auto-generated note. Although we propose a deep-learning-based solution, the current deep learning model is a data-driven model and the benchmark datasets with the annotation of visual entity extraction from slides are relatively small. This restrains the performance of the proposed deep learning framework, leading to that our framework may not work well in some unusual cases with an irregular layout structure, and this is a common problem for some other methods, such as [8, 69, 70, 116].

5.8 Summary

In this chapter, we have presented a novel educational video navigation tool that can effectively extract the hierarchical relationship between visual entities and associated visual entities with their corresponding speech texts based on their semantic similarity. In contrast to the existing approaches, our deep-learning-based method can extract effective features to differentiate different types of visual entities. Then, we have proposed a clustering approach to extract the hierarchical relationships of visual entities and align them with their corresponding descriptive speech texts. The experimental evaluation and user studies have suggested that, compared with the existing learning interfaces for navigating educational videos, our proposed approach is more effective and flexible for facilitating learning.

Chapter 6

Conclusion and Future Work

6.1 Conclusion

In this thesis, we have presented two works contributing to the development of bottom-up arbitrary-shape scene text detection and one application of scene text detection for navigating educational resources.

In Chapter 1, we have presented an overview of arbitrary-shape scene text detection and its applications in educational resource navigation, covering the latest breakthroughs and the merits and demerits of different genres and types of existing approaches. Secondly, we have provided our insights into some key issues in bottom-up arbitrary-shape scene text detection and its applications in educational resource navigation.

In Chapter 2, we have reviewed the existing works of two different types of modelling methods in scene text detection and the development of navigation systems that use text as the primary cue. After that, from Chapter 3 to Chapter 5, we have presented three of our representative works in addressing the bottom-up approaches in arbitrary-shape scene text detection and designed a novel navigation tool for educational resources.

In this chapter, we conclude the technical contributions of this thesis as follows.

The first contribution is related to the revitalization of GCN-based bottom-up arbitrary-shape scene text detection methods. We have analysed the main reasons why the existing bottom-up text detection methods are underperforming compared with the top-down methods even with the help of GCNs. To better depict the 'characterness' and 'streamline' of text, we have proposed false positive/negative suppression strategies that take visual-relational feature maps into account to make grouping

inference of densely designed text segments with regards to GCN's node classification and relational reasoning ability. A simple but effective shape-approximation module has been designed to replace the error-prone route-finding process in typical bottom-up methods. The state-of-the-art results obtained on several public datasets have demonstrated the effectiveness of our approach on arbitrary-shape text detection and further proved that the bottom-up methods are not inferior to, but surpass the top-down methods with the benefits of false positive/negative suppression in combination with shape-approximation.

The second contribution is related to tackling the error accumulation of false text segment detection and missing connection problems that prevent the bottom-up text detection approaches from achieving their great potential for handling arbitrary-shape text. We have introduced the deep morphology into this field for the first time in an effective way. Two deep morphological modules have been designed and embedded into the network to regularize the text segments based on their patterns of regularity learned through training. Extensive experiments conducted on four widely-used benchmark datasets have demonstrated the effectiveness of our proposed approach. The state-of-the-art results show that deep morphology can be used for the challenging task of arbitrary-shape text detection and its efficiency has also been demonstrated.

The third contribution is that we have presented a novel educational video navigation tool to effectively extract the hierarchical relationship between visual entities and associated visual entities with their corresponding speech texts based on their semantic similarity. Different from the existing approaches, our deep-learning-based method can extract effective features to differentiate three different types of visual entities including text, formula and graph. Then, we have proposed a clustering approach to restore hierarchical relationships of text entities before aligning them with their corresponding descriptive speech texts. The experimental evaluation and user studies have suggested that, compared with the existing learning interfaces for navigating educational videos, our proposed approach is more effective and flexible for facilitating learning and locating information.

6.2 Future Work

In this thesis, we proposed two bottom-up approaches toward accurate arbitrary-shape scene text detection and a novel navigation system using detected text as a cue for educational resources. There are still some challenging problems that need to be addressed. The potential directions for future work are listed as follows.

First, similar to most of the bottom-up arbitrary-shape scene text detection approaches, even with the help of deep learning technologies, the post-processing step still cannot be fully replaced. The empirical processing and heuristic processing also exist, for the binarization of text segmentation maps, filtering too small/low confidence detection results, selection of NMS threshold etc. However, selecting the best threshold for these tasks relies on human expertise, hindering the generalization ability of text detector. The automated machine learning (AutoML) techniques [140] should be used to optimize the threshold selection task above.

Second, text instances also have semantic features in addition to visual features. Designing the word spotting systems that do scene text detection and recognition together is an effective way to improve the performance of both tasks. The state-of-the-art language models [141, 142] should be introduced to build the modelling of words in the text instance so that the visual features and semantic features of text instances can be further fused.

Last but not least, the existing scene text detection methods and navigation systems can locate text instances, but the natural reading order cannot be effectively restored, resulting in loss of information integrity. This is especially common in educational document forms that contain dense text and tables. The restored reading order will also boost the performance of both scene text detection and recognition since most of the texts appear following the reading order. Recently, methods in [143, 144] tried to tackle the reconstruction of text blocks and reading order problems by language models. However, the layout structure of the text is also a strong visual cue that can be further explored to restore the natural reading order of the text instance.

Bibliography

[1] S.-X. Zhang, X. Zhu, J.-B. Hou, C. Liu, C. Yang, H. Wang, and X.-C. Yin, "Deep relational reasoning graph network for arbitrary shape text detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2020, pp. 9699–9708.

[2] J. Ye, Z. Chen, J. Liu, and B. Du, "Textfusenet: Scene text detection with richer fused features." Proc. Int. Joint Conf. Artif. Intell., 2020, pp. 516–522.

[3] B. Shi, X. Bai, and C. Yao, "An end-to-end trainable neural network for image-based sequence recognition and its application to scene text recognition," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 39, no. 11, pp. 2298–2304, 2016.

[4] A. Mishra, S. Shekhar, A. K. Singh, and A. Chakraborty, "Ocr-vqa: Visual question answering by reading text in images," in *Proc. IEEE Int. Conf. on Document Anal. and Recognit.* IEEE, 2019, pp. 947–952.

[5] F. Radenović, G. Tolias, and O. Chum, "Fine-tuning cnn image retrieval with no human annotation," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 41, no. 7, pp. 1655–1668, 2018.

[6] M. Dikmen and C. M. Burns, "Autonomous driving in the real world: Experiences with tesla autopilot and summon," in *Proc. Int. Conf. on Automot. User Interfaces and Interactive Veh. Appl.*

[7] G. M. Binmakhashen and S. A. Mahmoud, "Document layout analysis: A comprehensive survey," *ACM Computing Surveys (CSUR)*, vol. 52, no. 6, pp. 1–36, 2019.

[8] B. Zhao, S. Lin, X. Luo, S. Xu, and R. Wang, "A novel system for visual navigation of educational videos using multimodal cues," in *Proc. 25th ACM Int. Conf. Multimedia*, 2017, p. 1680–1688.

[9] C. K. Ch'ng and C. S. Chan, "Total-text: A comprehensive dataset for scene text detection and recognition," in *Proc. IEEE Int. Conf. on Document Anal. and Recognit.*, vol. 1. IEEE, 2017, pp. 935–942.

[10] Y. Liu, L. Jin, S. Zhang, C. Luo, and S. Zhang, "Curved scene text detection via transverse and longitudinal sequence connection," *Pattern Recognit.*, vol. 90, pp. 337–345, 2019.

[11] S. Long, J. Ruan, W. Zhang, X. He, W. Wu, and C. Yao, "Textsnake: A flexible representation for detecting text of arbitrary shapes," in *Proc. Eur. Conf. Comput. Vision*, 2018, pp. 20–36.

[12] X. Wang, Y. Jiang, Z. Luo, C.-L. Liu, H. Choi, and S. Kim, "Arbitrary shape scene text detection with adaptive text region representation," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2019, pp. 6449–6458.

[13] Y. Zhu, J. Chen, L. Liang, Z. Kuang, L. Jin, and W. Zhang, "Fourier contour embedding for arbitrary-shaped text detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2021, pp. 3123–3131.

[14] Y. Wang, H. Xie, Z.-J. Zha, M. Xing, Z. Fu, and Y. Zhang, "Contournet: Taking a further step toward accurate arbitrary-shaped scene text detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2020, pp. 11 753–11 762.

[15] S. Xiao, L. Peng, R. Yan, K. An, G. Yao, and J. Min, "Sequential deformation for accurate scene text detection," in *Proc. Eur. Conf. Comput. Vision*, 2020, pp. 108–124.

[16] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask r-cnn," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 2961–2969.

[17] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Proc. Eur. Conf. Comput. Vision*. Springer, 2016, pp. 56–72.

[18] Y. Baek, B. Lee, D. Han, S. Yun, and H. Lee, "Character region awareness for text detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2019, pp. 9365–9374.

[19] Z. Tian, M. Shu, P. Lyu, R. Li, C. Zhou, X. Shen, and J. Jia, "Learning shape-aware embedding for scene text detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2019, pp. 4234–4243.

[20] B. Shi, X. Bai, and S. Belongie, "Detecting oriented text in natural images by linking segments," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2017, pp. 2550–2558.

[21] F. Wang, Y. Chen, F. Wu, and X. Li, "Textray: Contour-based geometric modeling for arbitrary-shaped scene text detection," in *Proc ACM Int. Conf. Multimedia*, 2020, pp. 111–119.

[22] B. Shi, X. Bai, and S. Belongie, "Detecting oriented text in natural images by linking segments," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit. (CVPR)*, 2017, pp. 2550–2558.

[23] J. Ma, "Rrpn++: Guidance towards more accurate scene text detection," *arXiv preprint arXiv:2009.13118*, 2020.

[24] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *arXiv preprint arXiv:1506.01497*, 2015.

[25] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "Ssd: Single shot multibox detector," in *European conference on computer vision*. Springer, 2016, pp. 21–37.

[26] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779–788.

[27] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2015, pp. 3431–3440.

[28] C. Ma, L. Sun, Z. Zhong, and Q. Huo, "Relatext: exploiting visual relationships for arbitrary-shaped scene text detection with graph convolutional networks," *Pattern Recognit.*, vol. 111, p. 107684, 2021.

[29] C. Zhang, B. Liang, Z. Huang, M. En, J. Han, E. Ding, and X. Ding, "Look more than once: An accurate detector for text of arbitrary shapes," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2019, pp. 10 552–10 561.

[30] X. Wang, R. Girshick, A. Gupta, and K. He, "Non-local neural networks," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2018, pp. 7794–7803.

[31] J. Dai, H. Qi, Y. Xiong, Y. Li, G. Zhang, H. Hu, and Y. Wei, "Deformable convolutional networks," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 764–773.

[32] M. Liao, Z. Wan, C. Yao, K. Chen, and X. Bai, "Real-time scene text detection with differentiable binarization," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 07, 2020, pp. 11 474–11 481.

[33] P. Dai and X. Cao, "Comprehensive studies for arbitrary-shape scene text detection," *arXiv preprint arXiv:2107.11800*, 2021.

[34] H. Liu, A. Guo, D. Jiang, Y. Hu, and B. Ren, "Puzzlenet: scene text detection by segment context graph learning," *arXiv preprint arXiv:2002.11371*, 2020.

[35] W. Feng, W. He, F. Yin, X.-Y. Zhang, and C.-L. Liu, "Textdragon: An end-to-end framework for arbitrary shaped text spotting," in *Proc. IEEE Int. Conf. Comput. Vision*, 2019, pp. 9076–9085.

[36] J. Kim, P. J. Guo, C. J. Cai, S.-W. Li, K. Z. Gajos, and R. C. Miller, "Data-driven interaction techniques for improving navigation of educational videos," in *Proceedings of the 27th annual ACM symposium on User interface software and technology*, 2014, pp. 563–572.

[37] H. Yang and C. Meinel, "Content based lecture video retrieval using speech and video text information," *IEEE Trans. Learn. Technologies*, vol. 7, no. 2, pp. 142–154, 2014.

[38] A. Biswas, A. Gandhi, and O. Deshmukh, "Mmtoc: A multimodal method for table of content creation in educational videos," in *Proc. 23rd ACM Int. Conf. on Multimedia*, 2015, pp. 621–630.

[39] J. Adcock, M. Cooper, L. Denoue, H. Pirsiavash, and L. A. Rowe, "Talkminer: A lecture webcast search engine," in *Proc.18th ACM Int. Conf. Multimedia*, 2010, p. 241–250.

[40] C. Liu, J. Kim, and H.-C. Wang, "Conceptscape: Collaborative concept mapping for video learning," in *Proc. CHI Conf. Human Factors Comput. Syst.*, 2018, pp. 1–12.

[41] V. Balasubramanian, S. G. Doraisamy, and N. K. Kanakarajan, "A multimodal approach for extracting content descriptive metadata from lecture videos," *Journal of Intelligent Information Systems*, vol. 46, no. 1, pp. 121–145, 2016.

[42] B. Epshtein, E. Ofek, and Y. Wexler, "Detecting text in natural scenes with stroke width transform," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.* IEEE, 2010, pp. 2963–2970.

[43] W. Huang, Z. Lin, J. Yang, and J. Wang, "Text localization in natural images using stroke feature transform and text covariance descriptors," in *Proc. IEEE Int. Conf. Comput. Vision*, 2013, pp. 1241–1248.

[44] L. Neumann and J. Matas, "A method for text localization and recognition in real-world images," in *Asian Conf. Comput. Vision*. Springer, 2010, pp. 770–783.

[45] M. Liao, B. Shi, X. Bai, X. Wang, and W. Liu, "Textboxes: A fast text detector with a single deep neural network," in *Thirty-first AAAI conference on artificial intelligence*, 2017.

[46] M. Liao, B. Shi, and X. Bai, "Textboxes++: A single-shot oriented scene text detector," *IEEE Trans. Image Processing*, vol. 27, no. 8, pp. 3676–3690, 2018.

[47] J. Ma, W. Shao, H. Ye, L. Wang, H. Wang, Y. Zheng, and X. Xue, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Trans. Multimedia*, vol. 20, no. 11, pp. 3111–3122, 2018.

[48] Y. Jiang, X. Zhu, X. Wang, S. Yang, W. Li, H. Wang, P. Fu, and Z. Luo, "R2cnn: rotational region cnn for orientation robust scene text detection," *arXiv preprint arXiv:1706.09579*, 2017.

[49] W. He, X.-Y. Zhang, F. Yin, and C.-L. Liu, "Deep direct regression for multi-oriented scene text detection," in *Proceedings of the IEEE International Conference on Computer Vision*, 2017, pp. 745–753.

[50] A. Gupta, A. Vedaldi, and A. Zisserman, "Synthetic data for text localisation in natural images," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2016, pp. 2315–2324.

[51] X. Zhou, C. Yao, H. Wen, Y. Wang, S. Zhou, W. He, and J. Liang, "East: an efficient and accurate scene text detector," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2017, pp. 5551–5560.

[52] K.-H. Kim, S. Hong, B. Roh, Y. Cheon, and M. Park, "Pvanet: Deep but lightweight neural networks for real-time object detection," *arXiv preprint arXiv:1608.08021*, 2016.

[53] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie, "Feature pyramid networks for object detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2017, pp. 2117–2125.

[54] D. Deng, H. Liu, X. Li, and D. Cai, "Pixellink: Detecting scene text via instance segmentation," in *Proc. AAAI Conf. Artif. Intell.*, vol. 32, no. 1, 2018.

[55] D. He, X. Yang, W. Huang, Z. Zhou, D. Kifer, and C. L. Giles, "Aggregating local context for accurate scene text detection," in *Asian Conference on Computer Vision*. Springer, 2016, pp. 280–296.

[56] C. Wang, F. Yin, and C.-L. Liu, "Scene text detection with novel superpixel based character candidate extraction," in *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, vol. 1. IEEE, 2017, pp. 929–934.

[57] P. Lyu, M. Liao, C. Yao, W. Wu, and X. Bai, "Mask textspotter: An end-to-end trainable neural network for spotting text with arbitrary shapes," in *Proceedings of the European Conference on Computer Vision (ECCV)*, 2018, pp. 67–83.

[58] Z. Liu, G. Lin, S. Yang, F. Liu, W. Lin, and W. L. Goh, "Towards robust curve text detection with conditional spatial expansion," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2019, pp. 7269–7278.

[59] P. Dai, Y. Li, H. Zhang, J. Li, and X. Cao, "Accurate scene text detection via scale-aware data augmentation and shape similarity constraint," *IEEE Trans. Multimedia*, 2021.

[60] P. Dai, H. Zhang, and X. Cao, "Deep multi-scale context aware feature aggregation for curved scene text detection," *IEEE Trans. Multimedia*, vol. 22, no. 8, pp. 1969–1984, 2019.

[61] L. Qiao, S. Tang, Z. Cheng, Y. Xu, Y. Niu, S. Pu, and F. Wu, "Text perceptron: Towards end-to-end arbitrary-shaped text spotting," in *Proc. AAAI Conf. Artif. Intell.*, vol. 34, no. 07, 2020, pp. 11 899–11 907.

[62] P. Dai, S. Zhang, H. Zhang, and X. Cao, "Progressive contour regression for arbitrary-shape scene text detection," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2021, pp. 7393–7402.

[63] F. Yu, D. Wang, E. Shelhamer, and T. Darrell, "Deep layer aggregation," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2018, pp. 2403–2412.

[64] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *Proc. IEEE Int. Conf. Medical Image Comput. and Computer-assisted Intervention*. Springer, 2015, pp. 234–241.

[65] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, and S. Shao, "Shape robust text detection with progressive scale expansion network," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2019, pp. 9336–9345.

[66] S. Zhang, Y. Liu, L. Jin, Z. Wei, and C. Shen, "Opmp: An omnidirectional pyramid mask proposal network for arbitrary-shape scene text detection," *IEEE Trans. Multimedia*, vol. 23, pp. 454–467, 2020.

[67] W. Zhang, Y. Qiu, M. Liao, R. Zhang, X. Wei, and X. Bai, "Scene text detection with scribble line," in *Proc. IEEE Int. Conf. on Document Anal. and Recognit.* Springer, 2021, pp. 79–94.

[68] Y. Xu, Y. Wang, W. Zhou, Y. Wang, Z. Yang, and X. Bai, "Textfield: Learning a deep direction field for irregular scene text detection," *IEEE Trans. Image Processing*, vol. 28, no. 11, pp. 5566–5579, 2019.

[69] C. Nguyen and F. Liu, "Gaze-based notetaking for learning from lecture videos," in *Proc. ACM CHI Conf. on Human Factors in Comput. Syst.*, 2016, pp. 2093–2097.

[70] K. Yadav, A. Gandhi, A. Biswas, K. Shrivastava, S. Srivastava, and O. Deshmukh, "Vizig: Anchor points based non-linear navigation and summarization in educational videos," in *Proc. 21st Int. Conf. Intell. User Interfaces*, 2016, pp. 407–418.

[71] T.-J. K. P. Monserrat, S. Zhao, K. McGee, and A. V. Pandey, "Notevideo: facilitating navigation of blackboard-style lecture videos," in *Proc. ACM SIGCHI Conf. Human Factors in Comput. Syst.*, 2013, pp. 1139–1148.

[72] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," in *Proc. Int. Conf. Learn. Representat.*, May 2015, pp. 1–14.

[73] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2016, pp. 770–778.

[74] T. N. Kipf and M. Welling, "Semi-supervised classification with graph convolutional networks," in *Proc. Int. Conf. Learn. Representat.*, 2017, pp. 1–14.

[75] E. Xie, Y. Zang, S. Shao, G. Yu, C. Yao, and G. Li, "Scene text detection with supervised pyramid context network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 01, 2019, pp. 9038–9045.

[76] Z. Wang, L. Zheng, Y. Li, and S. Wang, "Linkage based face clustering via graph convolution network," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2019, pp. 1117–1125.

[77] A. Shrivastava, A. Gupta, and R. Girshick, "Training region-based object detectors with online hard example mining," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2016, pp. 761–769.

[78] D. Karatzas, L. Gomez-Bigorda, A. Nicolaou, S. Ghosh, A. Bagdanov, M. Iwamura, J. Matas, L. Neumann, V. R. Chandrasekhar, S. Lu *et al.*, "Icdar 2015 competition on robust reading," in *Proc. IEEE Int. Conf. on Document Anal. and Recognit.* IEEE, 2015, pp. 1156–1160.

[79] C. Yao, X. Bai, W. Liu, Y. Ma, and Z. Tu, "Detecting texts of arbitrary orientations in natural images," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.* IEEE, 2012, pp. 1083–1090.

[80] N. Nayef, F. Yin, I. Bizid, H. Choi, Y. Feng, D. Karatzas, Z. Luo, U. Pal, C. Rigaud, J. Chazalon *et al.*, "Icdar2017 robust reading challenge on multi-lingual scene text detection and script identification-rrc-mlt," in *Proc. IEEE Int. Conf. on Document Anal. and Recognit.*, vol. 1. IEEE, 2017, pp. 1454–1459.

[81] C. Yao, X. Bai, and W. Liu, "A unified framework for multioriented text detection and recognition," *IEEE Trans. Image Processing*, vol. 23, no. 11, pp. 4737–4749, 2014.

[82] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, and S. Shao, "Shape robust text detection with progressive scale expansion network," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2019, pp. 9336–9345.

[83] W. Wang, E. Xie, X. Song, Y. Zang, W. Wang, T. Lu, G. Yu, and C. Shen, "Efficient and accurate arbitrary-shaped text detection with pixel aggregation network," in *Proc. IEEE Int. Conf. Comput. Vision*, 2019, pp. 8440–8449.

[84] F. Wang, L. Zhao, X. Li, X. Wang, and D. Tao, "Geometry-aware scene text detection with instance transformation network," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2018, pp. 1381–1389.

[85] Y. Wang, H. Xie, Z. Zha, Y. Tian, Z. Fu, and Y. Zhang, "R-net: A relationship network for efficient and accurate scene text detection," *IEEE Trans. Multimedia*, vol. 23, pp. 1316–1329, 2020.

[86] E. Xie, Y. Zang, S. Shao, G. Yu, C. Yao, and G. Li, "Scene text detection with supervised pyramid context network," in *Proc. AAAI Conf. Artif. Intell.*, vol. 33, no. 01, 2019, pp. 9038–9045.

[87] R. M. Haralick, S. R. Sternberg, and X. Zhuang, "Image analysis using mathematical morphology," *IEEE Trans. Pattern Anal. and Mach. Intell.*, no. 4, pp. 532–550, 1987.

[88] C. Yi and Y. Tian, "Text string detection from natural scenes by structure-based partition and grouping," *IEEE Trans. Image Processing*, vol. 20, no. 9, pp. 2594–2605, 2011.

[89] Y. M. Hasan and L. J. Karam, "Morphological text extraction from images," *IEEE Trans. Image Processing*, vol. 9, no. 11, pp. 1978–1983, 2000.

[90] P. Shivakumara, T. Q. Phan, and C. L. Tan, "A laplacian approach to multi-oriented text detection in video," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 33, no. 2, pp. 412–419, 2010.

[91] K. Nogueira, J. Chanussot, M. Dalla Mura, W. Robson Schwartz, and J. A. dos Santos, "An introduction to deep morphological networks," *arXiv e-prints*, pp. arXiv–1906, 2019.

[92] R. Mondal, S. S. Mukherjee, S. Santra, and B. Chanda, "Morphological network: How far can we go with morphological neurons?" *arXiv preprint arXiv:1901.00109*, 2019.

[93] W. Wang, E. Xie, X. Li, W. Hou, T. Lu, G. Yu, and S. Shao, "Shape robust text detection with progressive scale expansion network," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit.*, 2019, pp. 9336–9345.

[94] C. Xu, W. Jia, T. Cui, R. Wang, Y.-f. Zhang, and X. He, "What's wrong with the bottom-up methods in arbitrary-shape scene text detection," *arXiv preprint arXiv:2108.01809*, 2021.

[95] G. Cavallaro, N. Falco, M. Dalla Mura, and J. A. Benediktsson, "Automatic attribute profiles," *IEEE Trans. Image Processing*, vol. 26, no. 4, pp. 1859–1872, 2017.

[96] Y. Tarabalka, J. Chanussot, and J. A. Benediktsson, "Segmentation and classification of hyperspectral images using watershed transformation," *Pattern Recognit.*, vol. 43, no. 7, pp. 2367–2379, 2010.

[97] J. Serra, "Image analysis and mathematical morphology," 1982.

[98] E. Zamora and H. Sossa, "Dendrite morphological neurons trained by stochastic gradient descent," *Neurocomputing*, vol. 260, pp. 420–431, 2017.

[99] R. Mondal, P. Purkait, S. Santra, and B. Chanda, "Morphological networks for image de-raining," in *International Conference on Discrete Geometry for Computer Imagery*. Springer, 2019, pp. 262–275.

[100] G. Franchi, A. Fehri, and A. Yao, "Deep morphological networks," *Pattern Recognit.*, vol. 102, p. 107246, 2020.

[101] R. Mondal, M. S. Dey, and B. Chanda, "Image restoration by learning morphological opening-closing network," *Mathematical Morphology-Theory and Applications*, vol. 4, no. 1, pp. 87–107, 2020.

[102] B. R. Vatti, "A generic solution to polygon clipping," *Communications of the ACM*, vol. 35, no. 7, pp. 56–63, 1992.

[103] Y. Liu, L. Jin, and C. Fang, "Arbitrarily shaped scene text detection with a mask tightness text detector," *IEEE Trans. Image Processing*, vol. 29, pp. 2918–2930, 2019.

[104] D. Shah, "By the numbers: Moocs in 2018," *classcentral.com. https://www.classcentral.com/report/mooc-stats-2018/ (accessed Dec. 1, 2021).*

[105] K. Yadav, K. Shrivastava, S. Mohana Prasad, H. Arsikere, S. Patil, R. Kumar, and O. Deshmukh, "Content-driven multi-modal techniques for non-linear video navigation," in *Proc. 20th Int. Conf. Intell. User Interfaces*, 2015, p. 333–344.

[106] P. J. Guo, J. Kim, and R. Rubin, "How video production affects student engagement: An empirical study of mooc videos," in *Proc. 1st ACM Conf. Learn. @ Scale Conf.*, 2014, p. 41–50.

[107] R. M. Felder, L. K. Silverman *et al.*, "Learning and teaching styles in engineering education," *Eng. Educ.*, vol. 78, no. 7, pp. 674–681, 1988.

[108] S. Sharma and A. Bumb, "The challenges faced in technology-driven classes during covid-19," *Int. Journal Distance Educ. Technologies*, vol. 19, no. 1, pp. 17–39, 2021.

[109] S. Dhawan, "Online learning: A panacea in the time of covid-19 crisis," *Journal of Educational Technol. Syst.*, vol. 49, no. 1, pp. 5–22, 2020.

[110] X. Che, H. Yang, and C. Meinel, "Automatic online lecture highlighting based on multimedia analysis," *IEEE Trans. Learn. Technologies*, vol. 11, no. 1, pp. 27–40, 2018.

[111] A. Smith, S. Leeman-Munk, A. Shelton, B. Mott, E. Wiebe, and J. Lester, "A multimodal assessment framework for integrating student writing and drawing in elementary science learning," *IEEE Trans. Learn. Technologies*, vol. 12, no. 1, pp. 3–15, 2018.

[112] H. Garcıa-Gonzalez, J. E. L. Gayo, and M. Paule-Ruiz, "Enhancing e-learning content by using semantic web technologies," *IEEE Trans. Learn. Technologies*, vol. 10, no. 4, pp. 544–550, 2017.

[113] P. Vrablecová and M. Šimko, "Supporting semantic annotation of educational content by automatic extraction of hierarchical domain relationships," *IEEE Trans. Learn. Technologies*, vol. 9, no. 3, pp. 285–298, 2016.

[114] R. Huang, D. Liu, A. Tlili, J. Yang, H. Wang *et al.*, "Handbook on facilitating flexible learning during educational disruption: The chinese experience in maintaining undisrupted learning in covid-19 outbreak," *Beijing: Smart Learn. Inst. Beijing Normal Univ.*, pp. 1–54, 2020.

[115] Y.-N. Chen, Y. Huang, S.-Y. Kong, and L.-S. Lee, "Automatic key term extraction from spoken course lectures using branching entropy and prosodic/semantic features," in *IEEE Spoken Language Technol. Workshop*, 2010, pp. 265–270.

[116] H. Jung, H. V. Shin, and J. Kim, "Dynamicslide: Exploring the design space of reference-based interaction techniques for slide-based lecture videos," in *Proc. Workshop Multimedia Accessible Human Comput. Interface*, 2018, p. 33–41.

[117] D. Augusto Borges Oliveira and M. Palhares Viana, "Fast cnn-based document layout analysis," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 1173–1180.

[118] S. A. Oliveira, B. Seguin, and F. Kaplan, "dhsegment: A generic deep-learning approach for document segmentation," in *Proc. 16th IEEE Int. Conf. Frontiers Handwriting Recognit.*, 2018, pp. 7–12.

[119] H. V. Shin, F. Berthouzoz, W. Li, and F. Durand, "Visual transcripts: Lecture notes from blackboard-style lecture videos," *ACM Trans. Graph.*, vol. 34, no. 6, Oct. 2015.

[120] K. Y. Wong, R. G. Casey, and F. M. Wahl, "Document anal. system," *IBM Journal of Research and Development*, vol. 26, no. 6, pp. 647–656, 1982.

[121] V. P. Le, N. Nayef, M. Visani, J.-M. Ogier, and C. D. Tran, "Text and non-text segmentation based on connected component features," in *Proc. 13th IEEE Int. Conf. on Document Anal. and Recognit. (ICDAR)*, 2015, pp. 1096–1100.

[122] D. Mahapatra, R. Mariappan, and V. Rajan, "Automatic hierarchical table of contents generation for educational videos," in *Companion Proc. Web Conf.*, 2018, p. 267–274.

[123] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.

[124] B. Shi, M. Yang, X. Wang, P. Lyu, C. Yao, and X. Bai, "Aster: An attentional scene text recognizer with flexible rectification," *IEEE Trans. Pattern Anal. and Mach. Intell.*, vol. 41, no. 9, pp. 2035–2048, 2019.

[125] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst. (NeurIPS)*, 2013, p. 3111–3119.

[126] J. Pennington, R. Socher, and C. Manning, "GloVe: Global vectors for word representation," in *Proc.Conf. Empirical Methods Natural Lang. Process.*, Doha, Qatar, Oct. 2014, pp. 1532–1543.

[127] P. Bojanowski, E. Grave, A. Joulin, and T. Mikolov, "Enriching word vectors with subword information," *arXiv preprint arXiv:1607.04606*, 2016.

[128] M. Kusner, Y. Sun, N. Kolkin, and K. Weinberger, "From word embeddings to document distances," in *Int. Conf. Machine Learning*, 2015, pp. 957–966.

[129] M. Bruls, K. Huizing, and J. J. Van Wijk, "Squarified treemaps," in *Data visualization 2000*. Springer, 2000, pp. 33–42.

[130] D. Karatzas, S. R. Mestre, J. Mas, F. Nourbakhsh, and P. P. Roy, "Icdar 2011 robust reading competition-challenge 1: reading text in born-digital images (web and email)," in *IEEE Int. Conf. Document Anal. and Recognit. (ICDAR)*, 2011, pp. 1485–1490.

[131] Y. Wu and P. Natarajan, "Self-organized text detection with minimal post-processing via border learning," in *Proc. IEEE Int. Conf. Comput. Vision*, 2017, pp. 5000–5009.

[132] M. Haurilet, Z. Al-Halah, and R. Stiefelhagen, "Spase-multi-label page segmentation for presentation slides," in *Proc. IEEE Winter Conf. on Appl. Comput. Vision*, 2019, pp. 726–734.

[133] A. Araujo, J. Chaves, H. Lakshman, R. Angst, and B. Girod, "Large-scale query-by-image video retrieval using bloom filters," *arXiv preprint*

arXiv:1604.07939, 2016.

[134] Z. Zhang, C. Zhang, W. Shen, C. Yao, W. Liu, and X. Bai, "Multi-oriented text detection with fully convolutional networks," in *Proc. IEEE Conf. Comput. Vision and Pattern Recognit. (CVPR)*, 2016, pp. 4159–4167.

[135] C. Xu, R. Wang, S. Lin, X. Luo, B. Zhao, L. Shao, and M. Hu, "Lecture2note: Automatic generation of lecture notes from slide-based educational videos," in *IEEE Int. Conf. Multimedia and Expo*, 2019, pp. 898–903.

[136] Z. Tian, W. Huang, T. He, P. He, and Y. Qiao, "Detecting text in natural image with connectionist text proposal network," in *Proc. Eur. Conf. Comput. Vision (ECCV)*. Springer, 2016, pp. 56–72.

[137] D. Deng, H. Liu, X. Li, and D. Cai, "Pixellink: Detecting scene text via instance segmentation," in *Proc. AAAI Conf. Artif. Intell.(AAAI)*, vol. 32, no. 1, 2018.

[138] E. Xie, Y. Zang, S. Shao, G. Yu, C. Yao, and G. Li, "Scene text detection with supervised pyramid context network," in *Proc. AAAI Conf. Artif. Intell.(AAAI)*, vol. 33, no. 01, 2019, pp. 9038–9045.

[139] C.-Y. Lin, "Rouge: A package for automatic evaluation of summaries," in *Text Summarization Branches Out*, 2004, pp. 74–81.

[140] X. He, K. Zhao, and X. Chu, "Automl: A survey of the state-of-the-art," *Knowledge-Based Systems*, vol. 212, p. 106622, 2021.

[141] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," *arXiv preprint arXiv:1911.02116*, 2019.

[142] T. B. Brown, B. Mann, N. Ryder, M. Subbiah, J. Kaplan, P. Dhariwal, A. Neelakantan, P. Shyam, G. Sastry, A. Askell *et al.*, "Language models are few-shot learners," *arXiv preprint arXiv:2005.14165*, 2020.

[143] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou, "Layoutlm: Pre-training of text and layout for document image understanding," in *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, 2020, pp. 1192–1200.

[144] Z. Wang, Y. Xu, L. Cui, J. Shang, and F. Wei, "Layoutreader: Pre-training of text and layout for reading order detection," *arXiv preprint arXiv:2108.11591*, 2021.