VSR: A Unified Framework for Document Layout Analysis combining Vision, Semantics and Relations

Peng Zhang[1], Can Li[1], Liang Qiao[1], Zhanzhan Cheng[2,1], Shiliang Pu[1], Yi Niu[1], and Fei Wu[2]

[1] Hikvision Research Institute, China
{zhangpeng23, lican9, qiaoliang6, chengzhanzhan, pushiliang.hri, niuyi}@hikvision.com
[2] Zhejiang University, China
wufei@cs.zju.edu.cn

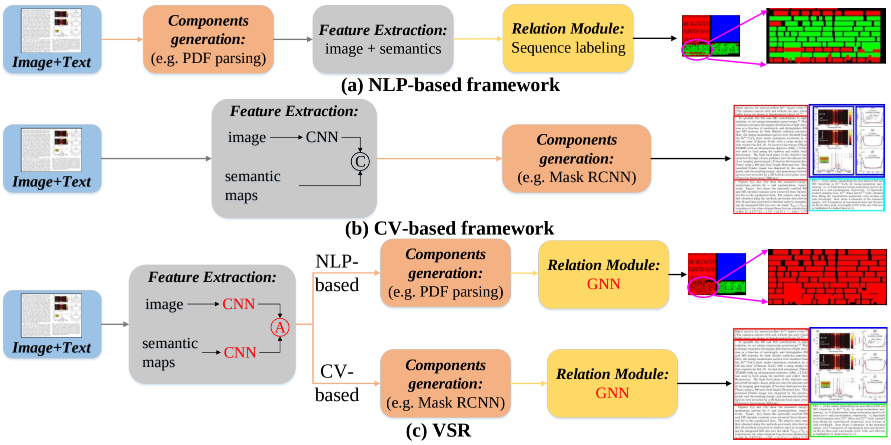
Abstract. Document layout analysis is crucial for understanding document structures. On this task, *vision* and *semantics* of documents, and *relations* between layout components contribute to the understanding process. Though many works have been proposed to exploit the above information, they show unsatisfactory results. NLP-based methods model layout analysis as a sequence labeling task and show insufficient capabilities in layout modeling. CV-based methods model layout analysis as a detection or segmentation task, but bear limitations of inefficient modality fusion and lack of relation modeling between layout components. To address the above limitations, we propose a unified framework VSR for document layout analysis, combining *vision*, *semantics* and *relations*. VSR supports both NLP-based and CV-based methods. Specifically, we first introduce *vision* through document image and *semantics* through text embedding maps. Then, modality-specific visual and semantic features are extracted using a *two-stream* network, which are *adaptively* fused to make full use of complementary information. Finally, given component candidates, a *relation module* based on graph neural network is incorported to model relations between components and output final results. On three popular benchmarks, VSR outperforms previous models by large margins. Code will be released soon.

Keywords: Vision · Semantics · Relations · Document layout analysis.

1 Introduction

Document layout analysis is a crucial step in automatic document understanding and enables many important applications, such as document retrieval [4], digitization [7] and editing. Its goal is to identify the regions of interest in unstructured document and recognize the role of each region. This task is challenging due to the diversity and complexity of document layouts.

Many deep learning models have been proposed on this task in both computer vision (CV) and natural language processing (NLP) communities. Most


Image+Text
Components generation: (e.g. PDF parsing)
Feature Extraction: image + semantics
Relation Module: Sequence labeling
Feature Extraction:
image → CNN
semantic maps
C
Components generation: (e.g. Mask RCNN)
Feature Extraction:
image → CNN
semantic maps → CNN
A
NLP-based
CV-based
Relation Module: GNN

(a) NLP-based framework

(b) CV-based framework

(c) VSR

Fig. 1. Comparison of multimodal document layout analysis frameworks. VSR supports both NLP-based and CV-based frameworks. Ⓒ and Ⓐ denote *concatenation* and *adaptive aggregation* fusion strategies. Different colored regions in the prediction results indicate different semantic labels (Paragraph, Figure, Figure Caption, Table Caption).

of them consider either only visual features [12,5,36,10,34,41,19,21] or only semantic features [6,17,27]. However, information from both modalities could help recognize the document layout better. Some regions (*e.g.,*Figure, Table) can be easily identified by visual features, while semantic features are important for separating visually similar regions (*e.g.,*Abstract and Paragraph). Therefore, some recent efforts try to combine both modalities [1,20,39,3]. Here we summarize them into two categories.

NLP-based methods (Fig 1 (a)) model layout analysis as a sequence labeling task and apply a bottom-up strategy. They first serialize texts into 1D token sequence[1]. Then, using both semantic and visual features (such as coordinates and image embedding) of each token, they determine token labels sequentially through a sequence labeling model. However, NLP-based methods show insufficient capabilities in layout modeling. For example in Fig. 1(a), all texts in a paragraph should have consistent semantic labels (Paragraph), but some of them are recognized as Figure Caption, which are the labels of adjacent texts.

CV-based methods (Fig 1 (b)) model layout analysis as object detection or segmentation task, and apply a top-down strategy. They first extract visual features by convolutional neural network and introduce semantic features through text embedding maps (at sentence-level [39] *or* character-level [3]), which are directly concatenated as the representation of document. Then, detection or

[1] In the rest of this paper, we assume text is available. There are tools available to extract text from PDF documents (*e.g.,* PDFMiner [28]) and document images (*e.g.,* OCR engine [30]).

segmentation models (*e.g.,* Mask RCNN [13]) are used to generate layout component candidates (coordinates and semantic labels). While capturing spatial information better compared to NLP-based methods, CV-based methods still have 3 limitations: (1) *limited semantics.* Semantic information are embedded in text at different granularities, including characters (or words) and sentences, which could help identify different document elements. For example, character-level features are better for recognizing components which need less context (*e.g.,* Author) while sentence-level features are better for contextual components (*e.g.,* Table caption). Exploiting semantics at one granularity could not achieve optimal performances. (2) *simple and heuristic modality fusion strategy.* Features from different modalities contribute differently to component recognition. Visual features contribute more to recognizing visually rich components (such as Figure and Table), while semantic features are better at distinguishing text-based components (Abstract and Paragraph). Simple and heuristic modality fusion by concatenation can not fully make use of complementary information between two modalities. (3) *lack of relation modeling between components.* Strong relations exist in documents. For example, “Figure” and “Figure Caption” often appear together, and “Paragraph”s have aligned bounding box coordinates. Such relations could be utilized to boost layout analysis performances.

In this paper, we propose a unified framework VSR for document layout analysis, combining ***V****ision*, ***S****emantics* and ***R****elation modeling*, as shown in Fig 1 (c). This framework can be applied to both NLP-based and CV-based methods. First, documents are fed into VSR in the form of images (vision) and text embedding maps (semantics at both character-level and sentence-level). Then, modality-specific visual and semantic features are extracted through a two-stream network, which are effectively combined later in a multi-scale adaptive aggregation module. Finally, a GNN(Graph Neural Network)-based relation module is incorporated to model relations between component candidates, and generate final results. Specifically, for NLP-based methods, text tokens serve as the component candidates and relation module predicts their semantic labels. While for CV-based methods, component candidates are proposed by detection or segmentation model (*e.g.,* Faster RCNN/ Mask RCNN) and relation module generates their refined coordinates and semantic labels.

Our work makes four key contributions:

- We propose a unified framework VSR for document layout analysis, combining *vision*, *semantics* and *relations* in documents.
- To exploit vision and semantics effectively, we propose a *two-stream* network to extract modality-specific visual and semantic features, and fuse them *adaptively* through an adaptive aggregation module. Besides, we also explore document semantics at different granularities.
- A GNN-based relation module is incorporated to model relations between document components, and it supports relation modeling in both NLP-based and CV-based methods.
- We perform extensive evaluations of VSR, and on three public benchmarks, VSR shows significant improvements compared to previous models.

2 Related Works

Document Layout Analysis. In this paper, we try to review layout analysis works from the perspective of modality used, namely, *unimodal layout analysis* and *multimodal layout analysis*.

Unimodal layout analysis exploits either only visual features [21,19] (document image) or only semantic features (document texts) to understand document structures. Using visual features, several works [5,36] have been proposed to apply CNN to segment various objects, *e.g.,* text blocks [10], text lines [34,18], words [41], figures or tables [12,29]. At the same time, there are also methods [6,17,27] which try to address the layout analysis problem using semantic features. However, all the above methods are strictly restricted to visual *or* semantic features, and thus are not able to exploit complementary information from other modalities.

Multimodal layout analysis tries to combine information from both visual and semantic modalities. Related methods can be further divided into two categories, NLP-based and CV-based methods. NLP-based methods work on low-level elements (*e.g.,* tokens) and model layout analysis as a sequence labeling task. MMPAN [1] is presented to recognize form structures. DocBank [20] is proposed as a large scale dataset of multimodal layout analysis and several NLP baselines have been released. However, the above methods show insufficient capabilities in layout modeling. CV-based methods introduce document semantics through text embedding maps, and model layout analysis as object detection or segmentation task. MFCN [39] introduces sentence granularity semantics and inserts the text embedding maps at the decision-level (end of network), while *dhSegment*^{T}[2] [3] introduces character granularity semantics and inserts text embedding maps at the input-level. Though showing great success, the above methods also bear the following limitations: limited semantics used, simple modality fusion strategy and lack of relation modeling between components.

To remedy the above limitations, we propose a unified framework VSR to exploit vision, semantics and relations in documents.

Two-stream networks. Two-stream networks are widely used to combine features in different modalities or representations [2] effectively. In action recognition, two-stream networks are used to capture the complementary *spatial* and *temporal* information [9]. In RGB-D saliency detection, the complete representations are fused from the deep features of the *RGB* stream and *depth* stream [11]. Also, two-stream networks are used to fuse different features of same input sample in sound event classification and image recognition [23]. Motivated by their successes, we apply two-stream networks to capture complementary *vision* and *semantics* information in documents.

Relation modeling. Relation modeling is a broad topic and has been studing for decades. In natural language processing, dependencies between sequential

[2] *dhSegment*^{T} means *dhSegment* with inputs of image and text embedding maps.

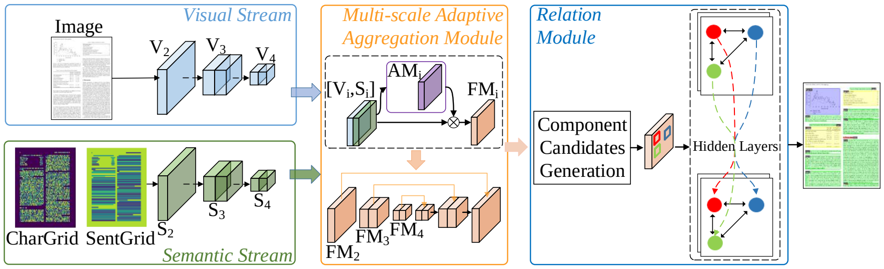


Fig. 2. The architecture overview of VSR. Best viewed in color.

texts are captured through *RNN* [15] or *Transformer* [32] architectures. In computer vision, non-local networks [35] and relation networks [16] are presented to model long-range dependencies between pixels and objects. Besides, in document image processing, relations between text and layout [38] or relations between document entities [40,24,42] are explored. As to multimodal layout analysis, NLP-based methods model it as a sequence labeling task and use *RNN* to capture component relations, while CV-based methods model it as object detection task but lack relation modeling between layout components. In this paper, we propose a GNN-based relation module, supporting relation modeling in both NLP-based or CV-based methods.

3 Methodology

3.1 Architecture Overview

Our proposed framework has three parts: two-stream ConvNets, a multi-scale adaptive aggregation module and a relation module (as shown in Fig 2). First, a two-stream convolutional network extracts modality-specific visual and semantic features, where visual stream and semantic stream take images and text embedding maps as input, respectively (Sec 3.2). Next, instead of simply concatenating the visual and semantic features, we aggregate them via a multi-scale adaptive aggregation module (Sec 3.3). Then, a set of component candidates are generated. Finally, a relation module is incorporated to model relations among those candidates and generate final results (Sec 3.4).

Notice that multimodal layout analysis can be modeled as sequence labeling (NLP-based methods) or object detection tasks (CV-based methods). Our framework supports both modeling types. The only difference is what the component candidates are and how to generate them. Component candidates are low-level elements (*e.g.,* text tokens) in NLP-based methods and can be generated by parsing PDFs, while candidates are high-level elements (regions) generated by detection or segmentation model (*e.g.,* Mask RCNN) in CV-based methods. In the rest of this paper, we will illustrate how VSR is applied to CV-based methods, and show it can be easily adapted to NLP-based methods in experiments on DocBank benchmark (Sec 4.3).

3.2 Two-stream ConvNets

CNN is known to be good at learning deep features. However, previous multimodal layout analysis works [39,3] only apply it to extract visual features. Text embedding maps are directly used as semantic features. This single-stream network design could not make full use of document semantics. Motivated by great success of two-stream network in various multimodal applications [9,23], we apply it to extract deep visual and semantic features.

Visual stream ConvNet. This stream directly takes document images as input and extracts multi-scale deep features using CNN backbones like ResNet [14]. Specifically, for an input image $x \in \mathbb{R}^{H \times W \times 3}$, multi-scale features maps (denoted by $\{V_2, V_3, V_4, V_5\}$) are extracted, where each $V_i \in \mathbb{R}^{\frac{H}{2^i} \times \frac{W}{2^i} \times C_i^V}$. H and W are the height and width of input image x, C_i^V is the channel dimension of feature map V_i, and $V_0 = x$.

Semantic stream ConvNet. Similar to [39,3], we introduce document semantics through text embedding maps $S_0 \in \mathbb{R}^{H \times W \times C_0^S}$, which are the input of semantic stream ConvNet. S_0 have same spatial sizes with document image x (V_0) and C_0^S denotes the initial channel dimension. This type of representation not only encodes text content, but also preserves the 2D layout of a document. Previously, only semantics at one granularity is used (character-level [3] *or* sentence-level[3] [39]). However, semantics at different granularities contribute to identification of different components. Thus, S_0 consists of both character and sentence level semantics. Next, we show how we build text embedding maps S_0.

The characters and sentences of a document page are denoted as $\mathbb{D}_c = \{(c_k, b_k^c) \,|k = 0, \cdots, n\}$ and $\mathbb{D}_s = \{(s_k, b_k^s) \,|k = 0, \cdots, m\}$, where n and m are the total number of characters and sentences. c_k and $b_k^c = (x_0, y_0, x_1, y_1)$ are the k-th character and its associated box, where (x_0, y_0) and (x_1, y_1) are top-left and bottom-right pixel coordinates. Similarly, s_k and b_k^s are the k-th sentence and its box location. Next, character embedding maps $CharGrid \in \mathbb{R}^{H \times W \times C_0^S}$ and sentence embedding maps $SentGrid \in \mathbb{R}^{H \times W \times C_0^S}$ can be constructed as follows.

$$CharGrid_{ij} = \begin{cases} E^c(c_k) & \text{if } (i,j) \in b_k^c \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

$$SentGrid_{ij} = \begin{cases} E^s(s_k) & \text{if } (i,j) \in b_k^s \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

[3] Sentence is a group of words or phrases, which usually ends with a period, question mark or exclamation point. For simplicity, we approximate it with text lines.

All pixels in each b_k^c (b_k^s) share the same character (sentence) embedding vector. E^c and E^s are the mapping functions of $c_k \to \mathbb{R}^{C_0^S}$ and $s_k \to \mathbb{R}^{C_0^S}$. In our implementation, E^c is a typical word embedding layer and we adopt pretrained language model BERT [8] as E^s. Finally, the text embedding maps S_0 can be constructed by applying LayerNorm normalization to the summation of $CharGrid$ and $SentGrid$, as shown in Eq.(3).

$$S_0 = LayerNorm\left(CharGrid + SentGrid\right) \tag{3}$$

Similar to the visual stream, semantic stream ConvNet then takes text embedding maps S_0 as input and extracts multi-scale features $\{S_2, S_3, S_4, S_5\}$, which have the same spatial sizes and channel dimension with $\{V_2, V_3, V_4, V_5\}$.

3.3 Multi-scale Adaptive Aggregation

Features from different modalities are important for identifying different objects. Modality fusion strategy should adaptively aggregate visual and semantic features. Thus, we design a multi-scale adaptive aggregation module that learns an attention map to combine visual features $\{V_2, V_3, V_4, V_5\}$ and semantic features $\{S_2, S_3, S_4, S_5\}$ adaptively. At scale i, this module first concatenates V_i and S_i, and then feed it into a convolutional layer to learn an attention map AM_i. Finally, aggregated multi-modal features FM_i is obtained. All operations in this module are formulated by:

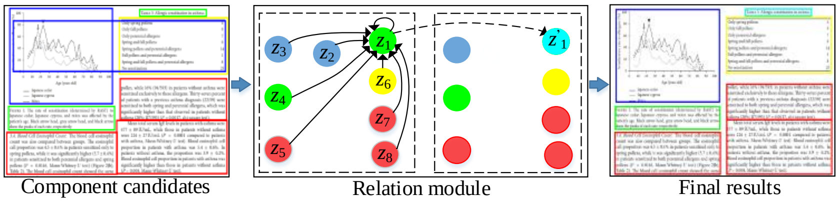
$$AM_i = h\left(g\left([V_i, S_i]\right)\right) \tag{4}$$

$$FM_i = AM_i \odot V_i + (1 - AM_i) \odot S_i \tag{5}$$

where $[\cdot]$ denotes the concatenation operation, $g\left(\cdot\right)$ is a convolutional layer with kernel size $1 \times 1 \times \left(C_i^V + C_i^S\right) \times C_i^S$ and $h\left(\cdot\right)$ is a non-linear activation function. $\odot$ denotes the element-wise multiplication. Through this module, a set of fused multi-modal features $FM = \{FM_2, FM_3, FM_4, FM_5\}$ are generated, which serve as the multimodal multi-scale features of a document. Then, FPN [22] (feature pyramid network) is applied on FM and provides enhanced representations.

3.4 Relation Module

Given aggregated features $FM = \{FM_2, FM_3, FM_4, FM_5\}$, a standard object detection or segmentation model (*e.g.,* Mask RCNN [26]) can be used to generate component candidates in a document. Previous works directly take those predictions as final results. However, strong relations exist between layout components. For example, bounding boxes of Paragraphs in the same column should be aligned; Table and Table Caption often appear together; there is no overlap between components. We find that such relations can be utilized to further refine predictions, as shown in Fig 3, *i.e.,* adjusting regression coordinates for aligned bounding boxes, correcting wrong prediction labels based on co-occurrence of



Component candidates Relation module Final results

Fig. 3. Illustration of relation module. It captures relations between component candidates, and thus improves detection results (remove false Figure prediction, correct Table Caption label and adjust Paragraph coordinates). The colors of semantic labels are: Figure, Paragraph, Figure Caption, Table, Table Caption. Best viewed in color.

components and removing false predictions based on non-overlapping property. Next, we show how we use GNN (graph neural network) to model component relations and how to use it to refine prediction results.

We represent a document as a graph $G = (O, E)$, where $O = \{o_1, \cdots, o_N\}$ is the node set and E is the edge set. Each node o_j represents a component candidate generated by the object detection model previously, and each edge represents the relation between two component candidates. Since remote regions in a document may also bear close dependencies (*e.g.*, a paragraph spans two columns), all regions constitute a neighbor relationship. Thus, the document graph is a fully-connected graph and $E \subseteq O \times O$. The key idea of our relation module is to update the hidden representations of each node by attending over its neighbors ($z_1, z_2, \cdots, z_8 \rightarrow z'_1$, as shown in Fig 3). With updated node features, we could predict its refined label and position coordinates.

Initially, each node, denoted by $o_j = (b_j, f_j)$, includes two pieces of information: position coordinates b_j and deep features $f_j = RoIAlign(FM, b_j)$. In order to incorporate both of them into node representation, we construct new node feature z_j as follows,

$$z_j = LayerNorm(f_j + e_j^{pos}(b_j)) \tag{6}$$

where $e_j^{pos}(b_j)$ is the position embedding vectors of j-th node.

Then, instead of explicitly specifying the relations between nodes, inspired by [33], we apply *self-attention* mechanism to automatically learn the relations, which has already shown great success in NLP and document processing [38,40,24,42]. Specifically, we adopt the popular scaled dot-product attention [32] to obtain sufficient expressive power. Scaled dot-product attention consists of queries Q and keys K of dimension d_k, and values V of dimension d_v. The output $\widehat{O}$ is obtained by weighted sum over all values in V, where the attention weights are obtained using Q and K, as shown in Eq.(7). Please refer to [32] for details.

$$\widehat{O} = Attention(Q, K, V) = softmax(\frac{QK^\top}{\sqrt{d_k}})V \tag{7}$$

In our context, node feature set $Z = \{z_1, \cdots, z_N\}$ serves as K, Q and V and updated node feature set $Z' = \{z'_1, \cdots, z'_N\}$ is the output $\widehat{O}$. We apply multi-head attention to further improve representation capacity of node features.

Finally, given updated node features Z', refined detection results of j-th node (j-th layout component candidate) $\widetilde{o_j} = \left(\widetilde{p^c_j}, \widetilde{b_j}\right)$ is computed as,

$$\widetilde{p^c_j} = Softmax(Linear_{cls}\left(z'_j\right)) \tag{8}$$

$$\widetilde{b_j} = Linear_{reg}\left(z'_j\right) \tag{9}$$

where $\widetilde{p^c_j}$ is the probability of belonging to c-th class, $\widetilde{b_j}$ is its refined regression coordinates. $Linear_{cls}$ and $Linear_{reg}$ are projection layers.

Relation module can be easily applied to NLP-based methods. In this case, node feature z_j in Eq.(6) is the representation of j-th low-level elements (*e.g.,* tokens). Then, GNN models pairwise relations between tokens and predicts their semantic labels ($\widetilde{p^c_j}$).

3.5 Optimization

Since multimodal layout analysis can be modeled as sequence labeling or object detection tasks, their optimization losses are different.

Layout analysis as sequence labeling. The loss function is formulated as,

$$\mathcal{L} = -\frac{1}{T}\sum_{j=1}^{T} log\ \widetilde{p_j}\left(y_j\right) \tag{10}$$

where, T is the number of low-level elements and y_j is the groundtruth semantic label of j-th element.

Layout analysis as object detection. The loss function is generated from two parts,

$$\mathcal{L} = \mathcal{L}_{DET} + \lambda\mathcal{L}_{RM} \tag{11}$$

where $\mathcal{L}_{DET}$ and $\mathcal{L}_{RM}$ are the losses used in candidate generation process and relation module. Both $\mathcal{L}_{DET}$ and $\mathcal{L}_{RM}$ consist of a cross entropy loss (classification) and a smooth L_1 loss (coordinate regression), as defined in [26]. Hyper-parameters λ controls the trade-off between two losses.

4 Experiments

4.1 Datasets

All three benchmarks provide document images and their original PDFs. Therefore, text could be directly obtained by parsing PDFs, allowing the explorations of multi-modal techniques. To compare with existing solutions on each benchmark, we use the same evaluation metrics as used by each benchmark.

Article Regions [31] consists of 822 document samples and 9 region classes are annotated (Title, Authors, Abstract, Body, Figure, Figure Caption, Table, Table Caption and References). The annotation is in object detection format and the evaluation metric is mean average precision (mAP).

PubLayNet [43] is a large-scale document dataset recently released by IBM. It consists of 360K document samples and 5 region classes are annotated (Text, Title, List, Figure, and Table). The annotation is also in object detection format. They use the same evaluation metric as used in the COCO competition, *i.e.,* the mean average precision (AP) @ intersection over union (IOU) [0.50:0.95].

DocBank [20] is proposed by Microsoft. It contains 500K document samples with 12 region classes (Abstract, Author, Caption, Equation, Figure, Footer, List, Paragraph, Reference, Section, Table and Title). It provides token-level annotations, and use F1 score as official evaluation metric. Also, it provides object detection annotations, supporting object detection method.

4.2 Implementation Details

Document image is directly used as input for visual stream. For semantic stream, we extract embedding maps (*SentGrid* and *CharGrid*) from text as input, where *SentGrid* is generated by pretrained BERT model [8] and *CharGrid* is obtained from a word embedding layer. They all have the same channel dimension size ($C_0^S = 64$). ResNeXt-101 [37] is used as backbone to extract both visual and semantic features (unless otherwise specified), which are later fused by a multi-scale adaptive aggregation and feature pyramid network.

For CV-based multimodal layout analysis methods, fused features are fed into RPN, followed by RCNN, to generate component candidates. In RPN, 7 anchor ratios (0.02, 0.05, 0.1, 0.2, 0.5, 1.0, 2.0) are adopted to handle document elements that vary in sizes and scales. In relation module, dimension of each candidate is set to 1024 and 2 layers of multi-head attention with 16 heads are used to model relations. We set λ in Eq.(11) to be 1 in all our experiments. For NLP-based multimodal layout analysis methods, low-level elements parsed from PDFs (*e.g.,* tokens) serve as component candidates, and relation module predicts their semantic labels.

Our model is implemented under the PyTorch framework. It is trained by the SGD optimizer with batchsize=2, momentum=0.9 and weight-decay=10^{-4}. The initial learning rate is set to 10^{-3}, which is divided by 10 every 10 epochs on Article Regions dataset and 3 epochs on the other two benchmarks. The training of model on Article Regions lasts for 30 epochs while on the other two benchmarks lasts for 6. All the experiments are carried out on Tesla-V100 GPUs. Source code will be released in the near future.

4.3 Results

Article Regions. We compare the performance of VSR on this dataset with two models: Faster RCNN and Faster RCNN *with context* [31]. Faster RCNN

Table 1. Performance comparisons on Article Regions dataset

Method	Title	Author	Abstract	Body	Figure	Figure Caption	Table	Table Caption	Reference	mAP
Faster RCNN [31]	-	1.22	-	87.49	-	-	-	-	-	46.38
Faster RCNN *w/ context* [31]	-	10.34	-	93.58	-	-	-	30.8	-	70.3
Faster RCNN *reimplement*	100.0	51.1	94.8	98.9	94.2	91.8	97.3	67.1	90.8	87.3
Faster RCNN *w/ context reimplement* [31]	100.0	60.5	90.8	98.5	**96.2**	91.5	**97.5**	64.2	91.2	87.8
VSR	**100.0**	**94**	**95**	**99.1**	95.3	**94.5**	96.1	**84.6**	**92.3**	**94.5**

Note: missing entries are because those results are not reported in their original papers.

Table 2. Performance comparisons on PubLayNet dataset.

Method	Dataset	Text	Title	List	Table	Figure	AP
Faster RCNN [43]	val	91	82.6	88.3	95.4	93.7	90.2
Mask RCNN [43]		91.6	84	88.6	96	94.9	91
VSR		**96.7**	**93.1**	**94.7**	**97.4**	**96.4**	**95.7**
Faster RCNN [43]	test	91.3	81.2	88.5	94.3	94.5	90
Mask RCNN [43]		91.7	82.8	88.7	94.7	95.5	90.7
DocInsightAI		94.51	88.31	94.84	95.77	97.52	94.19
SCUT		94.3	89.72	94.25	96.62	97.68	94.51
SRK		94.65	89.98	**95.14**	**97.16**	**97.95**	94.98
SiliconMinds		96.2	89.75	94.6	96.98	97.6	95.03
VSR		**96.69**	**92.27**	94.55	97.03	97.90	**95.69**

with context adds limited context (page numbers, region-of-interest position and size) as input in addition to document images.

In Table 1, we first show mAP as reported in their original papers [31]. For fair comparison, we reimplement those two models using the same backbone (ResNet-101) and neck configuration as used in VSR. We also report their performance after reimplementation. We can see that our reimplemented models have much higher mAP than their original models. We believe this is mainly because we use multiple anchor ratios in RPN, thus achieve better detection results on document elements with various sizes. VSR makes full use of vision, semantics and relations between components, showing highest mAP on most classes. On Figure and Table categories, VSR achieves comparable results and the slight performance drop will be further discussed in Sec 4.4.

PubLayNet. In Table 2, we compare the performance of VSR on this dataset with two pure image-based methods, Faster RCNN [26] and Mask RCNN [13]. While those two models present promising results (AP>90%) on validation dataset, VSR improves the performance on all classes and increases the final AP by 4.7%. VSR shows large performance improvements on text-related classes (Text, Title and List) since it also utilizes document semantics in addition to document image. On test dataset (also known as *leaderboard of ICDAR2021 layout analysis recognition competition*[4]), VSR surpasses all participating teams and ranks first, with 4.99% increase on AP compared with Mask RCNN baseline.

[4] https://icdar2021.org/competitions/competition-on-scientific-literature-parsing/

Table 3. Performance comparisons on DocBank dataset in F1 Score.

Method	Abstract	Author	Caption	Equation	Figure	Footer	List	Paragraph	Reference	Section	Table	Title	Macro Average
BERT_{base}	92.94	84.84	86.29	81.52	100.0	78.05	71.33	96.19	93.10	90.81	82.96	94.42	87.70
RoBERTa_{base}	92.88	86.18	89.44	82.48	100.0	80.14	73.53	96.46	93.41	93.37	83.89	95.11	88.91
LayoutLM_{base}	98.16	85.95	95.97	89.47	100.0	89.57	89.48	97.88	93.38	95.98	86.33	95.79	93.16
BERT_{large}	92.86	85.77	86.50	81.77	100.0	78.14	69.60	96.19	92.84	90.65	83.20	94.30	87.65
RoBERTa_{large}	94.79	87.24	90.81	83.70	100.0	83.92	74.51	96.65	93.34	94.07	84.94	94.61	89.88
LayoutLM_{large}	97.84	87.83	95.56	89.74	**100.0**	91.46	90.04	97.90	93.32	95.96	86.79	95.52	93.50
X101	97.17	82.27	94.35	89.38	88.12	90.29	90.51	96.82	87.98	94.12	83.53	91.58	90.51
$\text{X101+LayoutLM}_{base}$	98.15	89.07	**96.69**	94.30	99.90	92.92	93.00	98.43	94.37	96.64	88.18	95.75	94.78
$\text{X101+LayoutLM}_{large}$	98.02	89.64	96.66	94.40	99.94	93.52	92.93	98.44	94.30	96.70	88.75	95.31	94.88
VSR	**98.29**	**91.19**	96.32	**95.84**	99.96	**95.11**	**94.66**	**98.66**	**95.05**	**97.11**	**89.24**	**95.63**	**95.59**

Table 4. Performance comparisons on DocBank dataset in mAP.

Models	Abstract	Author	Caption	Equation	Figure	Footer	List	Paragraph	Reference	Section	Table	Title	mAP
Faster RCNN	96.2	88.9	93.9	**78.1**	85.4	**93.4**	86.1	67.8	89.9	76.7	77.2	**95.3**	86.3
VSR	**96.3**	**89.2**	**94.6**	77.3	**97.8**	93.2	**86.2**	**69.0**	**90.3**	**79.2**	**77.5**	94.9	**87.6**

DocBank. This dataset offers both token and detection annotations. Therefore, we could treat layout analysis task either as sequence labeling task or as object detection task, then compare VSR with existing solutions in both cases.

Layout analysis as sequence labeling. Using token-level annotations, we compare VSR with BERT [8], RoBERTa [25], LayoutLM [38], Faster RCNN with ResNeXt-101 [37] and ensemble models (ResNeXt-101+LayoutLM) in Table 3. Even though highest F1 score of Caption and Figure are achieved by ensemble model (ResNeXt-101+LayoutLM) and LayoutLM respectively, VSR achieves comparable results with small gaps ($\leq 0.37\%$). More importantly, VSR gets the highest scores on all other classes. This indicates that VSR is significantly better than BERT, RoBERTa and LayoutLM architectures on document layout analysis task.

Layout analysis as object detection. Since both VSR and Faster RCNN with ResNeXt-101 can provide object detection results, we further compare them in object detection format using mAP as evaluation metric. Results in Table 4 show that VSR outperforms Faster RCNN on most classes, except Equation, Footer and Title. Overall, VSR shows 1.3% gains in final mAP.

4.4 Ablation Studies

VSR introduces multi-granularity semantics, two-stream network with adaptive aggregation, and relation module. Now we explore how each of them contributes to VSR's performance improvement on Article Regions dataset.

Effects of multi-granularity semantic features. To understand whether multi-granularity semantic features indeed improve VSR's performance, we compare 4 versions of VSR (*vision-only*, *vision+character*, *vision+sentence*, *vision+ character+sentence*) in Table 5. Here *character* and *sentence* refer to semantic features at two different granularities. We can see that, introducing document

Table 5. Effects of semantic features at different granularities.

Vision	Semantics		Title	Author	Abstract	Body	Figure	Figure Caption	Table	Table Caption	Reference	mAP
	Char	Sentence										
√			100.0	51.1	94.8	98.9	94.2	91.8	97.3	67.1	90.8	87.3
√	√		100.0	71.4	**96.5**	98.9	95.6	**93.6**	96.9	68.6	89.9	90.2
√		√	100.0	60.2	95.5	**99.0**	**97.8**	93.2	98.9	**73.0**	91.2	89.8
√	√	√	**100.0**	**84.3**	96.1	98.7	95.7	92.5	**99.4**	71.4	**92.4**	**92.3**

Table 6. Effects of two-stream network with adaptive aggregation.

Method		Title	Author	Abstract	Body	Figure	Figure Caption	Table	Table Caption	Reference	mAP	FPS
Single-stream at input level	R101	94.7	58.7	82.7	98.1	97.9	**96.3**	91.8	63.7	91.5	86.2	19.07
	R152	100.0	50.5	85.3	97.9	**98.0**	94.4	93.3	62.6	90.5	85.8	18.15
Single-stream at decision level	R101	99.5	67.6	95.1	98.8	95.0	93.2	96.6	70.7	91.3	89.8	**19.79**
	R152	100.0	80.2	91.0	**99.4**	96.0	92.4	98.3	**73.8**	91.7	91.4	16.43
VSR	R101	**100.0**	**84.3**	**96.1**	98.7	95.7	92.5	**99.4**	71.4	**92.4**	**92.3**	13.94

semantics at each granularity alone can boost analysis performance while combining both of them leads to highest mAP. This is consistent with how humans comprehend documents. Humans can better recognize regions which require little context from characters/words (*e.g.,*Author) and those which need context from sentences (*e.g.,*Table caption).

Effects of two-stream network with adaptive aggregation. We propose a two-stream network with adaptive aggregation module to combine vision and semantics of document. To verify its effectiveness, we compare our VSR with its multimodal single-stream counterparts in Table 6. Instead of using extra stream to extract semantic features, single-stream networks directly use text embedding maps and concatenate them with visual features at input-level [3] or decision-level [39]. [3] performs concatenation fusion in the input level and shows worse performances, while [39] fuses multimodal features in the decision level and achieves impressive performances (89.8 mAP). VSR first extracts visual and semantic features separately using two-stream network, and then fuses them adaptively. This leads to highest mAP (92.3). At the same time, VSR can run at real-time (13.94 frames per second). We also experiment on larger backbone (ResNet-152) and reach consistent conclusions as shown in Table 6.

Effects of relation module. To verify the effectiveness of relation module (*RM*), we compare two versions of Faster RCNN and VSR in Table 7, *i.e.,* with *RM* and without *RM.* Since both labels and position coordinates can be refined in *RM*, both unimodal Faster RCNN and VSR show consistent improvements after incorporating relation module, with 5.3% and 2.2% increase respectively. Visual examples are given in Fig 4. However, for Figure component, performance may slightly drop after introducting *RM.* The reason is that, while removing duplicate predictions, our relation module may also risk removing correct pre-

Table 7. Effects of relation module.

Method		Title	Author	Abstract	Body	Figure	Figure caption	Table	Table caption	Reference	mAP
Faster RCNN	w/o RM	1	51.1	94.8	98.9	**94.2**	91.8	97.3	67.1	90.8	87.3
	w/ RM	**1**	**88.4**	**99.1**	**99.1**	85.4	**92.6**	**98.0**	**79.2**	**91.6**	**92.6**
VSR	w/o RM	1	84.3	**96.1**	98.7	**95.7**	92.5	**99.4**	71.4	**92.4**	92.3
	w/ RM	**1**	**94**	95	**99.1**	95.3	**94.5**	96.1	**84.6**	92.3	**94.5**

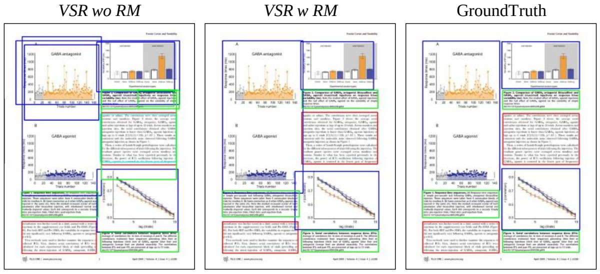


Fig. 4. Qualitative comparison between *VSR w/wo RM*. Introducing *RM* effectively removes duplicate predictions and provides more accurate detection results (both labels and coordinates). The colors of semantic labels are: Figure, Body, Figure Caption.

dictions. But still, we see improvements on overall performances, showing the benefits of introducting relations.

Limitations. As mentioned above, in addition to document images, VSR also requires the positions and contents of texts in the document. Therefor, the generalization of VSR may be not good enough compared with its unimodal counterparts, which we'll address in the future.

5 Conclusion

In this paper, we present a unified framework VSR for multimodal layout analysis combining vision, semantics and relations. We first introduce semantics of document at character and sentence granularities. Then, a two-stream convolutional network is used to extract modality-specific visual and semantic features, which are further fused in the adaptive aggregation module. Finally, given component candidates, a relation module is adopted to model relations between them and output final results. On three benchmarks, VSR outperforms its unimodal and multimodal single-stream counterparts significantly. In the future, we will investigate pre-training models with VSR and extend it to other tasks, such as information extraction.

References

1. Aggarwal, M., Sarkar, M., Gupta, H., Krishnamurthy, B.: Multi-modal association based grouping for form structure extraction. In: WACV. pp. 2064–2073 (2020)
2. Baltrusaitis, T., Ahuja, C., Morency, L.: Multimodal machine learning: A survey and taxonomy. IEEE Trans. Pattern Anal. Mach. Intell. **41**(2), 423–443 (2019)
3. Barman, R., Ehrmann, M., Clematide, S., Oliveira, S.A., Kaplan, F.: Combining visual and textual features for semantic segmentation of historical newspapers. CoRR **abs/2002.06144** (2020)
4. BinMakhashen, G.M., Mahmoud, S.A.: Document layout analysis: A comprehensive survey. ACM Comput. Surv. **52**(6), 109:1–109:36 (2020)
5. Chen, K., Seuret, M., Liwicki, M., Hennebert, J., Ingold, R.: Page segmentation of historical document images with convolutional autoencoders. In: ICDAR. pp. 1011–1015 (2015)
6. Conway, A.: Page grammars and page parsing. A syntactic approach to document layout recognition. In: ICDAR. pp. 761–764 (1993)
7. Corbelli, A., Baraldi, L., Grana, C., Cucchiara, R.: Historical document digitization through layout analysis and deep content classification. In: ICPR. pp. 4077–4082 (2016)
8. Devlin, J., Chang, M., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: NAACL-HLT. pp. 4171–4186 (2019)
9. Feichtenhofer, C., Pinz, A., Zisserman, A.: Convolutional two-stream network fusion for video action recognition. In: CVPR. pp. 1933–1941 (2016)
10. Gatos, B., Louloudis, G., Stamatopoulos, N.: Segmentation of historical handwritten documents into text zones and text lines. In: ICFHR. pp. 464–469 (2014)
11. Han, J., Chen, H., Liu, N., Yan, C., Li, X.: Cnns-based RGB-D saliency detection via cross-view transfer and multiview fusion. IEEE Trans. Cybern. **48**(11), 3171–3183 (2018)
12. He, D., Cohen, S., Price, B.L., Kifer, D., Giles, C.L.: Multi-scale multi-task FCN for semantic page segmentation and table detection. In: ICDAR. pp. 254–261 (2017)
13. He, K., Gkioxari, G., Dollár, P., Girshick, R.B.: Mask R-CNN. In: ICCV. pp. 2980–2988 (2017)
14. He, K., Zhang, X., Ren, S., Sun, J.: Deep residual learning for image recognition. In: CVPR. pp. 770–778 (2016)
15. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural Comput. **9**(8), 1735–1780 (1997)
16. Hu, H., Gu, J., Zhang, Z., Dai, J., Wei, Y.: Relation networks for object detection. In: CVPR. pp. 3588–3597 (2018)
17. Krishnamoorthy, M.S., Nagy, G., Seth, S.C., Viswanathan, M.: Syntactic segmentation and labeling of digitized pages from technical journals. IEEE Trans. Pattern Anal. Mach. Intell. **15**(7), 737–747 (1993)
18. Lee, J., Hayashi, H., Ohyama, W., Uchida, S.: Page segmentation using a convolutional neural network with trainable co-occurrence features. In: ICDAR. pp. 1023–1028 (2019)
19. Li, K., Wigington, C., Tensmeyer, C., Zhao, H., Barmpalios, N., Morariu, V.I., Manjunatha, V., Sun, T., Fu, Y.: Cross-domain document object detection: Benchmark suite and method. In: CVPR. pp. 12912–12921 (2020)
20. Li, M., Xu, Y., Cui, L., Huang, S., Wei, F., Li, Z., Zhou, M.: Docbank: A benchmark dataset for document layout analysis. In: COLING. pp. 949–960 (2020)

21. Li, X., Yin, F., Xue, T., Liu, L., Ogier, J., Liu, C.: Instance aware document image segmentation using label pyramid networks and deep watershed transformation. In: ICDAR. pp. 514–519 (2019)
22. Lin, T., Dollár, P., Girshick, R.B., He, K., Hariharan, B., Belongie, S.J.: Feature pyramid networks for object detection. In: CVPR. pp. 936–944 (2017)
23. Lin, T., RoyChowdhury, A., Maji, S.: Bilinear CNN models for fine-grained visual recognition. In: ICCV. pp. 1449–1457 (2015)
24. Liu, X., Gao, F., Zhang, Q., Zhao, H.: Graph convolution for multimodal information extraction from visually rich documents. In: NAACL-HLT. pp. 32–39 (2019)
25. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized BERT pretraining approach. CoRR **abs/1907.11692** (2019)
26. Ren, S., He, K., Girshick, R.B., Sun, J.: Faster R-CNN: towards real-time object detection with region proposal networks. In: NeurIPS. pp. 91–99 (2015)
27. Shilman, M., Liang, P., Viola, P.A.: Learning non-generative grammatical models for document analysis. In: ICCV. pp. 962–969 (2005)
28. Shinyama, Y.: Pdfminer: Python pdf parser and analyzer. Retrieved on **11** (2015)
29. Siegel, N., Lourie, N., Power, R., Ammar, W.: Extracting scientific figures with distantly supervised neural networks. In: JCDL. pp. 223–232 (2018)
30. Smith, R.: An overview of the tesseract OCR engine. In: ICDAR. pp. 629–633 (2007)
31. Soto, C., Yoo, S.: Visual detection with context for document layout analysis. In: EMNLP-IJCNLP. pp. 3462–3468 (2019)
32. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, L., Polosukhin, I.: Attention is all you need. In: NeurIPS. pp. 5998–6008 (2017)
33. Velickovic, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., Bengio, Y.: Graph attention networks. In: ICLR (2018)
34. Vo, Q.N., Lee, G.: Dense prediction for text line segmentation in handwritten document images. In: ICIP. pp. 3264–3268 (2016)
35. Wang, X., Girshick, R.B., Gupta, A., He, K.: Non-local neural networks. In: CVPR. pp. 7794–7803 (2018)
36. Wick, C., Puppe, F.: Fully convolutional neural networks for page segmentation of historical document images. In: DAS. pp. 287–292 (2018)
37. Xie, S., Girshick, R.B., Dollár P., He, K.: Aggregated residual transformations for deep neural networks. In: CVPR. pp. 5987–5995 (2017)
38. Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M.: Layoutlm: Pre-training of text and layout for document image understanding. In: KDD. pp. 1192–1200 (2020)
39. Yang, X., Yumer, E., Asente, P., Kraley, M., Kifer, D., Giles, C.L.: Learning to extract semantic structure from documents using multimodal fully convolutional neural networks. In: CVPR. pp. 4342–4351 (2017)
40. Yu, W., Lu, N., Qi, X., Gong, P., Xiao, R.: PICK: processing key information extraction from documents using improved graph learning-convolutional networks. In: ICPR. pp. 4363–4370 (2020)
41. Zagoris, K., Pratikakis, I., Gatos, B.: Segmentation-based historical handwritten word spotting using document-specific local features. In: ICFHR. pp. 9–14 (2014)
42. Zhang, P., Xu, Y., Cheng, Z., Pu, S., Lu, J., Qiao, L., Niu, Y., Wu, F.: TRIE: end-to-end text reading and information extraction for document understanding. In: MM. pp. 1413–1422 (2020)
43. Zhong, X., Tang, J., Jimeno-Yepes, A.: Publaynet: Largest dataset ever for document layout analysis. In: ICDAR. pp. 1015–1022 (2019)