XYLayoutLM: Towards Layout-Aware Multimodal Networks For Visually-Rich Document Understanding

Zhangxuan Gu[1,2], Changhua Meng[2], Ke Wang[2], Jun Lan[2], Weiqiang Wang[2], Ming Gu[2], Liqing Zhang[1*]

[1]MoE Key Lab of Artificial Intelligence, Department of Computer Science and Engineering,
Shanghai Jiao Tong University, [2]Ant Group

zhangxgu@126.com

{changhua.mch,kaywang.wk,yelan.lj,weiqiang.wwq,guming.mg}@antgroup.com

zhang-lq@cs.sjtu.edu.cn

Abstract

Recently, various multimodal networks for Visually-Rich Document Understanding(VRDU) have been proposed, showing the promotion of transformers by integrating visual and layout information with the text embeddings. However, most existing approaches utilize the position embeddings to incorporate the sequence information, neglecting the noisy improper reading order obtained by OCR tools. In this paper, we propose a robust layout-aware multimodal network named XYLayoutLM to capture and leverage rich layout information from proper reading orders produced by our Augmented XY Cut. Moreover, a Dilated Conditional Position Encoding module is proposed to deal with the input sequence of variable lengths, and it additionally extracts local layout information from both textual and visual modalities while generating position embeddings. Experiment results show that our XYLayoutLM achieves competitive results on document understanding tasks.

1. Introduction

While significant progress has been made in natural language processing and visual understanding [5, 7, 8, 20], less attention has been paid to their challenging variant in the multimodal document understanding domain. The Visually-Rich Document Understanding (VRDU) [28] task requires combining the abundant image, text, and layout information from scanned/digital-born documents (images, PDFs, *etc.*) Such technology can benefit a great variety of scenarios such as report/receipt understanding, automatical form filling, and document relation extraction. As a result, it is in great need of effective and efficient VRDU approaches.

To this end, researchers have developed sophisticated pipelines for tackling this task [2, 10, 16, 18, 28–30]. Generally speaking, early attempts can be divided into the categories of textual-based [4, 6, 10], convolution-based [12, 15, 24, 26, 34] and GCN-based [19] methods. Text-based methods, *e.g.*, XLM-RoBERT [6] and InfoXLM [4], usually rely on the representation ability of self-supervised models like Bert [7] pretrained on large datasets. Convolution-based method Chargrid [15] utilized a fully convolutional network that predicted a segmentation mask and bounding boxes for document representation. More recently, [19] introduces a Graph Convolutional Networks based model to fuse the textual and visual feature from scanned documents.

Although attempts like LayoutLM [28], LayoutLMv2 [30] and LayoutXLM [29] have been made to tackle document understanding in a multimodal manner, they still confront two limitations: (1) They rely on the tokens and boxes from OCR [31] tools without exploring the effect of reading orders. The proper reading orders refer to the well-organized readable token sequences, which may not be unique. Intuitively, the reading order of input tokens is crucial to many tasks such as language translation [27] and VQA [33]. For example, the meaning of a sentence may be changed when we shuffle the words, resulting in mistakes during language translation. A common solution is to use position embeddings to denote such sequential order of input tokens. However, we find that multimodal models with widely-used relative position embeddings still suffer improper reading order. Proper reading orders implicitly include the layout information, which is essentially needed in VRDU tasks. (2) They usually leverage fixed-length absolute/relative position embeddings in transformers. Once the model is trained, it can not deal with the test data with longer token sequences. Although bilinear interpolation on position embeddings can be applied to the longer sequence, the performance is not satisfying. Recently, Conditional Position Encoding
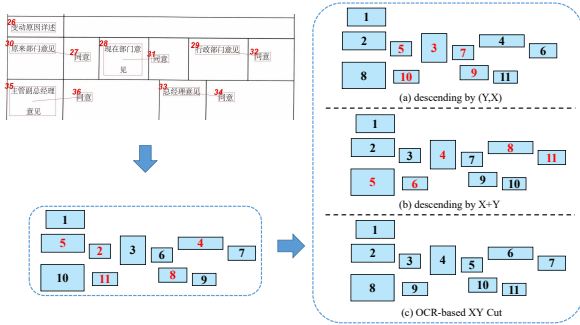
---

*Corresponding author.

Figure 1. An example from XFUN dataset. The reading order is denoted by the indices in the boxes while the red ones mean their orders are improper.

(CPE) [5] is proposed to deal with inputs of variable lengths in image classification tasks. It reshaped the input tokens to 2D features and dynamically extracted local neighbor context from input tokens with convolutions. However, since CPE is designed for only visual tokens, it can not handle 1D textual tokens in VRDU tasks.

In this paper, we propose an improved version of LayoutLMv2 [30], **XYLayoutLM**. Instead of pretraining on large private/public document understanding datasets, XYLayoutLM focuses on the generation of position embeddings with two under-explored limitations in VRDU, *i.e.*, improper reading orders, and the disability of dealing with longer sequence, as mentioned above.

Although it seems a fundamental requirement of multi-modal tasks to have proper reading orders, it is non-trivial to directly obtain such reading orders from documents due to various formats, *e.g.*, tables, and columns. Specifically, we show a form from XFUN [29] dataset in Figure 1. The default reading order is noisy. Based on the boxes obtained by OCR tools, traditional sorting approaches such as arranging the tokens in a top-to-bottom and left-to-right way are not satisfying. For example, we list two simple heuristic rules in this figure, namely (a) descending first by Y-axis then X-axis, (b) descending by Y+X conditioned on the left-top points of the token boxes. However, the red indices in Figure 1 still highlight the tokens with improper reading orders. Finally, we utilize the XY Cut [11] (c) and successfully obtain one proper reading order. Interestingly, some tokens in the same row may have different locations due to the noise in OCR recognition. It fails two heuristic rules which need the accuracy location of tokens. However, we can still obtain a series of proper reading orders for training by our proposed Augmented XY Cut as an augmentation strategy.

For input sequences of variable lengths, we utilize a novel Dilated Conditional Position Encoding (DCPE) module to adaptively generate position embeddings according to their input lengths with the dilated convolutions for ex-

tracting local layouts. We demonstrate that the XYLayoutLM can lead to better performance than previous LayoutLMs [28–30], which will benefit a great variety of real-world document understanding applications. We summarize our contributions as follows.

- For the first time, Augmented XY Cut is proposed and utilized to sort the input tokens for generating different proper reading orders in VRDU tasks. It extracts and leverages the layout information to achieve competitive performances.

- To deal with input sequences of variable lengths, we propose a Dilated Conditional Position Encoding as the position embedding generator to adaptively process the 1D textual and 2D visual tokens. Benefitting from proper reading orders, DCPE can further extract rich local layouts of input tokens with dilated convolutions.

- Comprehensive experiments are conducted on VRDU datasets. Our XYLayoutLM achieves competitive performance among all listed VRDU approaches on semantic entity recognition and relation extraction tasks.

2. Related Works

2.1. Visually-Rich Document Understanding

Recently, transformers-based methods have been proved to be effective on many computer vision [5, 8, 20] and natural language process [4, 7, 10] domains. Among them, [28–30] proposed a series of transformer-based models focusing on VRDU tasks. As our baseline, LayoutXLM [28] is the multilingual version of LayoutLMv2 [30]. They achieved impressive results by successfully combining textual, layout, and visual features. However, those methods may feed the input tokens to the transformer in the improper reading order caused by OCR tools on complex documents, which will harm VRDU performance. In this paper, we pay more attention to the under-explored challenge, *i.e.*, the proper reading order of input tokens, which is significant to the model performance.

2.2. Positional Encoding

Positional encodings are commonly employed to incorporate the order of sequences because self-attention is permutation-equivalent. Existing research can be grouped into two categories: absolute and relative position encodings. When the transformer-based model was first proposed by [27], they designed a delicate sin-cos function as the absolute positional encoding. After that, [7] used a learnable absolute embedding which is an embedding of the same length to the input sequence. It can be jointly updated with the network weights during training. Recently, by considering the distance between tokens, [25] proposed to change

position embedding from absolute way into the relative way. However, they can not handle the longer sequences with fixed-length position encodings. To this end, Conditional Position Encoding (CPE) [5] was proposed to deal with the input sequence of variable lengths in the image classification task. It generates position embeddings conditioned on the local context extracted by 2D convolutional layers. However, CPE can not be used in multimodal networks due to the 1D text features in document understanding tasks.

2.3. Reading Order Detection

Reading order detection [1, 3, 9, 17, 21, 22] aims to capture proper reading orders for documents. Generally speaking, humans tend to read documents left-to-right and up-to-bottom ways. However, such simple sorting rules may fail due to the tokens extracted by OCR tools on complex documents. Recently, [35] proposed a multimodal network for reading order detection with a large benchmark made by tremendous complex documents. However, compared to our method, the labor for collecting 500k standard Word files and the time for training a LayoutReader [35] cannot be ignored. Meanwhile, the inference time for LayoutReader on reading order detection is much longer than our method(see Appendix). In this paper, we proposed a simple yet effective augmentation algorithm based on XY Cut [11] to obtain different proper reading orders.

3. Methodology

3.1. Overview

The overall XYlayoutLM architecture is depicted in Figure 2. The model takes the images, textual tokens, and text locations (boxes) as the input. Visual tokens are obtained by adaptively pooling the feature map of ResNeXt-101 to 7×7. Then we flatten and concentrate it with the textual tokens to form the input token sequence following [29]. Two individual position embedding generators are utilized for encoding the tokens into position embeddings and box embeddings. Different from the baseline model LayoutXLM [29], our XYLayoutLM has two advantages: (1) An Augmented XY Cut module is proposed to sort the input tokens for different proper reading orders, which will contribute to the model performances by leveraging the layout information in reading order. (2) Instead of generating position embeddings with fixed-length MLPs, we propose the Dilated Conditional Position Encoding (DCPE) module to handle the input tokens of variable lengths from texts and images. In this section, we will first briefly introduce LayoutXLM [29] and then elaborate on the above-proposed components.

3.2. Review of LayoutXLM

Recall that LayoutXLM [29] accepts inputs of three modalities: text, image, and layout (*i.e.*, token locations).

The input of each modality is converted to an embedding sequence by a fixed-length MLP operated on the position indices as shown in Figure 2. The text and image embeddings are concatenated, plus the layout embedding to get the input embedding. After that, the input embeddings are encoded by a transformer with the spatial-aware self-attention mechanism within and between modalities. Finally, the visual/text token representations outputted by the transformer are used in the document understanding tasks. Since the architecture of self-attention layers is not our main concern, we omit it here and refer readers to [29, 30] for the details.

3.3. Proper Reading Orders

How to obtain proper reading orders of documents like forms and receipts is an open question. Intuitively, it is possible to infer how token boxes are aligned and where significant horizontal and vertical gaps are present from projection profiles. Hence, the projection profiles of token boxes can be used to determine the reading order. In this section, we first introduce the projection profiles of token boxes and then present the Augmented XY Cut algorithm.

Projection Profiles. Suppose we are given a set of token boxes $B = \{b_i\}_{i=1}^{K}$, where each $b_i = [x_1^i, y_1^i, x_2^i, y_2^i] \in \mathbb{Z}^4$ denotes a box and K is the number of OCR extracted tokens. We also define the minimum and maximum token locations in B as (x_{min}, y_{min}) and (x_{max}, y_{max}). Then the horizontal mapping H_{b_i} of box b_i is formulated as a indicative function:

$$H_{b_i}(y) = \begin{cases} 1, & y_1^i \leq y \leq y_2^i \\ 0, & otherwise \end{cases} \quad (1)$$

where $y \in \mathbb{Z}^{[y_{min}, y_{max}]}$. For a location y on Y-axis, $H_{b_i}(y)$ effectively mean whether y is in the projection interval $[y_1^i, y_2^i]$. Based on H_{b_i}, we can define the horizontal projection profile of the set B by summing all horizontal mapping functions of individual boxes:

$$H_B(y) = \sum_{i=1}^{K} H_{b_i}(y). \quad (2)$$

The values of $H_B(y)$ represent how many token boxes are projected onto the Y-axis that covers the input variable y.

Similar to $H_B(y)$, the vertical projection profile of B can be denoted as follows:

$$V_B(x) = \sum_{i=1}^{K} V_{b_i}(x), \quad (3)$$

where

$$V_{b_i}(x) = \begin{cases} 1, & x_1^i \leq x \leq x_2^i \\ 0, & otherwise \end{cases} \quad (4)$$

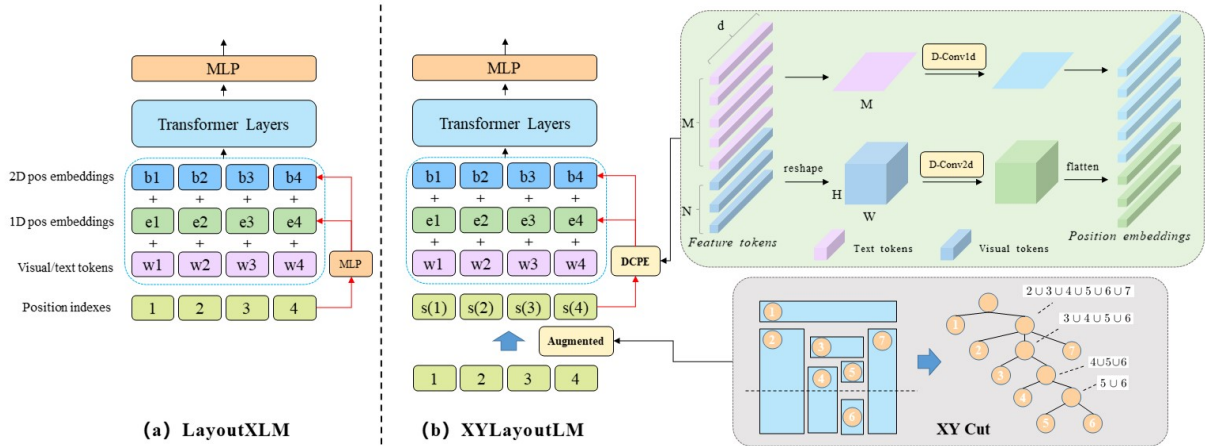is the vertical mapping on b_i with $x \in \mathbb{Z}^{[x_{min}, x_{max}]}$.

Figure 2. The overview of **XYLayoutLM**. Different from LayoutXLM, our XYLayoutLM proposes Augmented XY Cut and DCPE to extract and utilize layout information for multimodal document understanding. Best viewed in Adobe Acrobat DC.

Valleys in Projection Profiles. For simplicity, let us take the horizontal projection profile H_B as an example. As we mentioned before, after we have projected the token boxes B to Y-axis to get corresponding intervals $\{[y_1^i, y_2^i]\}_{i=1}^K$, $H_B(y)$ is like a histogram for counting the intervals that cover y. As a result, there might be some valleys in the histogram. The valley here is defined as $y^* \in \mathbb{Z}^{[y_{min}, y_{max}]}$ that meet the condition $H_B(y^*) = 0$. There is no token box in the valleys. Thus, valleys of projection profiles can determine where the division has to take place.

Augmented XY Cut Algorithm. Traditional XY Cut is a heuristic divide and conquer algorithm first proposed by [23] to segment a sentence into words according to the values of every pixel. In 1995, [11] utilized it to decompose documents (like newspapers) by applying XY Cut on the connected components. However, no current work explores the XY Cut in multimodal models or other deep learning transformers for obtaining proper reading orders.

As mentioned in the introduction, the token locations are often recognized with noise. Instead of expensive human annotations of reading orders, we propose an augmentation strategy on these noisy locations to generate different proper reading orders during network training. We believe the proper reading orders implicitly contain important layout information, significantly benefiting document understanding tasks. We only perform traditional XY Cut as a pre-process during the inference stage.

With the projection profiles and valleys defined above, Augmented XY Cut can be explained as follows. To better introduce it, we construct an XY Tree for recording the reading order while performing Augmented XY Cut. As shown in Algorithm 1 and Figure 2(b), we take the boxes set B as input, and the algorithm will output an index array $O = \{s(i)\}_{i=1}^K$ as one proper reading order. At first,

we create a root node without any index. Then, unlike traditional XY Cut, we propose an augmentation strategy based on three thresholds: $\lambda_x, \lambda_y, \theta$. Specifically, $\lambda_x(\lambda_y)$ are thresholds that determine whether we shift a box on X-axis (Y-axis). If so, we will shift a box with $\theta \cdot v_x (\theta \cdot v_y)$ pixels in one direction. For every box, we generate two random values v_x, v_y from $N(-1, 1)$. If $|v_x| > \lambda_x(|v_y| > \lambda_y)$, then this box will be shifted with $\theta \cdot v_x(\theta \cdot v_y)$ pixels with the direction according to the sign of $v_x(v_y)$. In this paper, we set hyper-parameters as $0.5, 0.5, 5$ by default according to the experimental results.

At each step, the projection profiles are calculated in either horizontal or vertical directions. Then a division is performed at the valleys in the corresponding projection profile to obtain several clusters. Their token boxes are gathered as the new child nodes of XY Tree in descending order. To obtain a proper reading order of length K is now divided into sub-tasks in each cluster with the number of boxes as the sequence length. The process is repeated recursively until no sufficient valleys are left in both profiles. If a cluster has more than one box and it can not be divided by both horizontal and vertical projection profiles, then the reading order inside this cluster will follow the heuristic rules, *e.g.*, descending first by Y-axis then X-axis. Finally, the output reading order is obtained by collecting the indices on leaves of the XY Tree *w.r.t* the tree height.

To better explain the generation of XY Tree, we take Figure 2(b) as an example. In the first step, we horizontally project all seven boxes to the Y-axis by calculating the values of horizontal projection profile $H_B(y)$. We find there is only one valley and then perform a division to get two clusters in descending order according to their location on Y-axis (see 1 and 2∪3∪4∪5∪6∪7 in Figure 2(b)). The first cluster only has one element, *i.e.* box 1, and thus it is the

Algorithm 1 Augmented XY Cut Algorithm

Require: boxes: $B = \{b_i\}_{i=1}^K$, thresholds: $\lambda_x, \lambda_y, \theta$
Ensure: proper reading order: $O = \{s(i)\}_{i=1}^K$
 1: Create a root node.   ▷ Init XY Tree
 2: Do augmentation on B with $\lambda_x, \lambda_y, \theta$. ▷ Augmentation
 3: Find the valleys of horizontal (H_B) or vertical (V_B) projection profiles.
 4: Do divisions at valleys. Whenever divisions are made, create a new child node. At each recursion level, horizontal and vertical divisions alternate.
 5: Do Step 3-4 recursively until no further divisions are possible.
 6: Gather the indices on leaves as output O.

---

first leaf of XY Tree. The second cluster has six elements with the candidate order array $2 \cup 3 \cup 4 \cup 5 \cup 6 \cup 7$ and thus is fed to the second step with vertical projection profiles. In the second step, two valleys are detected, and thus, the order of 2 and 7 are decided as the leaves of XY Tree while $3 \cup 4 \cup 5 \cup 6$ still need further divisions. By iteratively performing horizontal and vertical projections, we can obtain the final reading orders on the tree leaves. The pseudo-code is shown in the Appendix.

3.4. Dilated Conditional Position Encoding

Conditional Position Encoding (CPE) [5] aims to generate a various-length position embedding for different inputs in image classification task. Specifically, it reshapes the flattened input sequence X back to X' in the 2D visual space. Then, convolutional layers are repeatedly applied to the X' to produce the positional embedding E with proper kernel and padding size to keep the resolution. Finally, the position embedding E is flattened and added to the token embeddings as the transformer input.

However, simply replacing the MLP in LayoutXLM to CPE reduces the performance in VRDU tasks. One reason is the wrong neighbors of input tokens due to the improper reading order. Since CPE conditions on local context with convolutions, wrong neighbors will harm the model performances. Another reason is that CPE is designed specialized for image classification. The input visual tokens of image classification are 16×16 patches, and they can be naturally reshaped to 2D for local context extraction. However, in multimodal tasks, we also have 1D textual tokens in our input. These textual tokens only have 1D relations, so that they can not be reasonably reshaped to 2D.

The first problem is solved with a proper reading order obtained by using our Augmented XY Cut. In this section, we propose Dilated Conditional Position Encoding (DCPE) to tackle the second problem, *i.e.*, how to extract 1D local layouts from texts. As shown in Figure 2(a), our DCPE processes the textual and visual features individually. Specif-

ically, DCPE reshapes the 2D visual features and generates their position embeddings following the CPE. While for textual features, we utilized 1D convolutions to extract 1D local layouts. The encoded embeddings from the texts and images are concentrated as the final output.

Another observation is that multimodal tasks often need larger receptive fields while capturing local layouts. For example, in the sentence "he is a very handsome boy", the relation of "he" and "boy" is essential but can not be successfully captured by standard 1D convolutions due to the small convolution kernel size (*e.g.*, 3). To this end, we adopt dilated convolution [32] to replace standard convolutions, aiming for long-range neighbor information with larger receptive fields. Let l be the dilation rate and the dilated convolution $*_l$ can be formulated as:

$$(F *_l k)(\mathbf{p}) = \sum_{\mathbf{s}+l\mathbf{t}=\mathbf{p}} F(\mathbf{s})k(\mathbf{t}) \tag{5}$$

where F, k are the input feature map and filter. By repeatedly stacking dilated convolutions with different dilation rates $l > 1$, the DCPE module will pay more attention to the long-range neighbor information. Besides, the dilated convolutions have the same parameters as standard convolutions given the same kernel size, which means our DCPE will not increase the model complexity. Note that new parameters in DCPE are initialized by Xavier and updated with the whole model during training.

4. Experiments

4.1. Setup

Datasets. Following LayoutXLM [29], we conduct experiments on widely used VRDU datasets FUNSD [14] and XFUN [29]. FUNSD is a form understanding dataset for scanned documents. It contains 199 annotated forms with 31485 words. XFUN is a benchmark for multilingual Form Understanding by extending the FUNSD to 7 other languages, including Chinese, Japanese, Spanish, French, Italian, German, and Portuguese, with 1393 fully annotated forms. Each language includes 199 forms, where the training set includes 149 forms, and the test set includes 50 forms. These two datasets provide the official OCR annotations (bounding boxes and tokens) as the input.

Tasks. We focus on two tasks from VRDU, Semantic Entity Recognition (SER) and Relation Extraction (RE). Specifically, SER assigns each token a semantic label from a set of four predefined categories: question, answer, header, or other. For RE, following [29], we construct the set of relation candidates by generating all possible pairs of input tokens. We utilize a specific embedding layer for every pair to generate token type embedding as the token relation representation. The representations of head and tail are concatenated and fed into a bi-affine classifier. The F1 score is

Task	Methods	XFUN Avg.	ZH	JA	ES	FR	IT	DE	PT
	XLM-RoBERT [6]	0.7047	0.8774	0.7761	0.6105	0.6743	0.6687	0.6814	0.6818
	InfoXLM [4]	0.7207	0.8868	0.7865	0.6230	0.7015	0.6751	0.7063	0.7008
SER	LayoutXLM [29]	0.8056	0.8924	0.7921	0.7550	0.7902	0.8082	0.8222	0.7903
	LayoutXLM+CPE	0.8047	0.8776	0.7909	0.7551	0.7908	0.8063	0.8227	0.7898
	XYLayoutLM	**0.8204**	**0.9176**	**0.8057**	**0.7687**	**0.7997**	**0.8175**	**0.8335**	**0.8001**
	XLM-RoBERT [6]	0.4769	0.5105	0.5800	0.5295	0.4965	0.5305	0.5041	0.3982
	InfoXLM [4]	0.4910	0.5214	0.6000	0.5516	0.4913	0.5281	0.5262	0.4170
RE	LayoutXLM [29]	0.6432	0.7073	0.6963	0.6896	0.6353	0.6415	0.6551	0.5718
	LayoutXLM+CPE	0.6399	0.7059	0.6968	0.6812	0.6238	0.6399	0.6474	0.5723
	XYLayoutLM	**0.6779**	**0.7445**	**0.7059**	**0.7259**	**0.6521**	**0.6572**	**0.6703**	**0.5898**

Table 1. Comparison with different methods on the XFUN *w.r.t* F1 score (↑), where "SER" denotes the semantic entity recognition and "RE" denotes the relation extraction.

Methods	Modality	SER
BERT [7]	Language	0.6026
RoBERTa [6]	Language	0.6648
BROS [13]	Language	0.8121
LayoutLMv1 [28]	Language + Layout+ Vision	0.7927
LayoutXLM [29]	Language + Layout+ Vision	0.8276
DocFormer [2]	Language + Layout+ Vision	0.8334
SelfDoc [18]	Language + Layout+ Vision	0.8336
StructuralLM* [16]	Language + Layout	**0.8514**
XYLayoutLM	Language + Layout+ Vision	0.8335

Table 2. Comparison with different methods on the FUNSD *w.r.t* F1 score (↑). The * means StructuralLM use the LARGE model while others use BASE models.

Methods	SER		RE	
	ZH	ES	ZH	ES
LayoutXLM	0.8924	0.7550	0.7073	0.6896
+ CPE	0.8776	0.7306	0.7059	0.6812
+ 2×CPE	0.8819	0.7412	0.7082	0.6820
+ 2×DCPE	0.8952	0.7548	0.7097	0.6843
+ XY Cut	0.8903	0.7562	0.7281	0.7175
+ Aug XY Cut	0.9023	0.7570	0.7389	0.7213
+ Aug XY Cut & CPE	0.9037	0.7597	0.7401	0.7236
+ Aug XY Cut & DCPE	**0.9176**	**0.7687**	**0.7445**	**0.7259**

Table 3. Ablation studies of **XYLayoutLM** on XFUN (Chinese, English) for SER and RE tasks *w.r.t* F1 score (↑). 2× means we use two convolutional layers in this module.

Methods	SER		RE	
	ZH	ES	ZH	ES
default order	0.8924	0.7550	0.7073	0.6896
remove-pos-embed	0.8842	0.7477	0.6941	0.6682
descending (Y,X)	0.8857	0.7486	0.7297	0.7179
descending (X,Y)	0.8561	0.7343	0.6858	0.6549
descending (X+Y)	0.8844	0.7513	0.7235	0.7086
XY Cut	0.8903	0.7562	0.7281	0.7175
Aug descending (Y,X)	0.8925	0.7543	0.7331	0.7212
Aug XY Cut (0.5, 0.5, 1)	0.8913	0.7568	0.7282	0.7178
Aug XY Cut (0.2, 0.2, 5)	0.9011	0.7586	0.7387	0.7202
Aug XY Cut (0.5, 0.5, 5)	**0.9023**	**0.7600**	**0.7389**	**0.7213**
Aug XY Cut (0.7, 0.7, 5)	0.8918	0.7541	0.7260	0.7166
Aug XY Cut (0.5, 0.5, 10)	0.8702	0.7399	0.6920	0.6894

Table 4. F1 scores (↑) of baseline **LayoutXLM** based on different reading orders on XFUN (Chinese, English).

XFUN, the learning rate and batch size are set as 5×10^{-5} and 32 for the SER task, respectively. We train the model with 1000 iterations for convergence. For the RE task, the batch size is 8 with 2500 iterations for training. While on the FUNSD, the batch size is 16, and we train the model for 1000 iterations following [30].

4.2. Main Results

Here we compare our method with textual-based methods XLM-RoBERT [6], InfoXLM [4] and LayoutXLM [29] on XFUN. The results are shown in Table 1. From the table we can observe that XYLayoutLM achieves the best performance among the listed methods. More specifically, among the multimodal methods, XYLayoutLM outperforms the original LayoutXLM [29] by **1.48%** F1 score on the XFUN dataset for the SER task. Besides, our XYLayoutLM achieves a 0.6779 F1 score in RE task, which is an obvious improvement beyond the baseline LayoutXLM (0.6432). Similar conclusions are drawn on the FUNSD dataset as shown in Table 2. Our XYLayoutLM achieves

used as the evaluation metric for both two tasks.

Model variants. We init the weight of our XYLayoutLM with pretrained LayoutXLM$_{BASE}$. We set hidden size $d = 768$ and use a 12-layer 12-head transformer. The visual backbone is ResNeXt101-FPN, and the visual features are from its P_2 layer in FPN following LayoutXLM.

Training details. We use the same hyper-parameters with LayoutXLM for fair comparisons on two datasets. On

DCPE		SER		RE	
Text	Image	ZH	ES	ZH	ES
Conv2d	Conv2d	0.9037	0.7597	0.7059	0.6812
Conv1d	Conv1d	0.9091	0.7613	0.7066	0.6832
Conv1d	Conv2d	0.9140	0.7625	0.7256	0.7106
D-Conv1d	Conv2d	0.9163	0.7669	0.7440	0.7244
Conv1d	D-Conv2d	0.9149	0.7642	0.7427	0.7211
D-Conv1d	D-Conv2d	**0.9176**	**0.7687**	**0.7445**	**0.7259**

Table 5. F1 scores (↑) of **XYLayoutLM** based on different DCPE architectures on XFUN (Chinese, English).

comparable performances with latest methods like Doc-Former [2] and SelfDoc [18]. Note that StructuralLM* [16] uses the LARGE model to get the best performance while other methods in this table only use the BASE model.

Another observation is that only using CPE [5] on the LayoutXLM for position embeddings generation harms the performances on all tasks. This observation verifies our claim on the weakness of CPE for multimodal networks as mentioned in Section 3.4.

4.3. Ablation Studies

We perform SER experiments on the Chinese and English subsets of the XFUN dataset for the ablation studies. First, we show the impact of progressively integrating our different components: the DCPE and Augmented XY Cut module, to the baseline in Table 3. Then we explore different settings of each component individually.

Analysis for Components. As shown in Table 3, we first use the CPE to generate position embeddings instead of MLP in LayoutXLM, which decreases about 0.2% F1 score. It can be explained that the local context obtained by CPE for position embeddings generation is noisy due to the unreasonable reading order and wrong neighbors. In the third and fourth rows, we replace the CPE with our proposed DCPE with dilated convolutions, resulting in an improvement on all tasks. Note that $2\times$ means we stack two convolutional layers in each module since only one dilated convolutions may lose the information in the holes.

However, the performance gain of only using DCPE suffers from improper reading orders. Next, we only add XY Cut to the baseline, leading to significant improvements of F1 score on both SER (1%) and RE (3%) tasks, which verifies the essential effect of proper reading order. Furthermore, when we only perform Augmented XY Cut (Aug XY for short in table), its improvement is satisfactory. The last two rows show the performances of CPE and DCPE associated with Augmented XY Cut. We can observe that our DCPE achieves much better results than CPE because of the larger receptive fields on both textual and visual features.

Moreover, benefiting from the Augmented XY Cut, the improvements of DCPE are highly promoted. When

adapted to the baseline model in the default improper reading order, DCPE only gains 0.1% F1 score improvement on the Chinese subset of XFUN in the SER task. However, after Augmented XY Cut, the improvement comes to 1.5%.

In total, the whole improvements of XYLayoutLM upon baseline LayoutXLM indicates the effectiveness of our two contributions, Augmented XY Cut and DCPE.

Analysis for Augmented XY Cut. The token reading order is an essential factor of the effective document understanding method. Thus, to evaluate the improvement achieved by our proposed Augmented XY Cut, we conduct experiments on XFUN with different reading orders based on the baseline LayoutXLM as shown in Table 4. When we removed all the position embeddings as shown in the second row, the performance decreased for SER and RE tasks, which indicates the importance of position embeddings for incorporating the reading order. The next four rows are heuristic rules for sorting tokens, *i.e.*, descending first by Y-axis then X-axis, first by X-axis then Y-axis, by Y+X based on the left-top point of the token box and traditional XY Cut. However, their performances are not satisfying compared to the baseline. Finally, by using our Augmented XY Cut, the model achieves the best performance. Note that we set the hyper-parameter $\lambda_x, \lambda_y, \theta$ as $0.5, 0.5, 5$ since it has slightly better performance than other choices.

Analysis for DCPE. Besides the ability to deal with various-length inputs, the DCPE module also plays a critical role in our XYLayoutLM for gathering local layouts from both textual and visual features. Thus, we compare several settings inside the DCPE module to improve its effectiveness. As presented in Table 5, with the replacement of standard convolutional layers to dilated ones for textual and visual tokens, the network performance improves steadily and achieves the peak F1 score when using dilated convolutions for both textual and visual tokens. Another observation is that 1D convolutions can better extract textual features than 2D ones, which also verifies our claim on the reasons for the failure of CPE in multimodal networks.

With these ablation studies, we conclude that in XYLayoutLM: the Augmented XY Cut and DCPE module all play essential roles *w.r.t.* the final performance.

4.4. Visualizations

Effects on attention scores. We have shown that XYLayoutLM can have better performance than the original baseline LayoutXLM. However, because the Augmented XY Cut and DCPE provide the layout information implicitly in the position embeddings, it is interesting to see the attention weights of the transformers. Given a document, the size of the attention score is 561×561 following LayoutXLM (512 textual tokens and 49 visual tokens). We visualize the attention score matrix from different attention layers of one sample with the same reading order in Figure 3. Note that
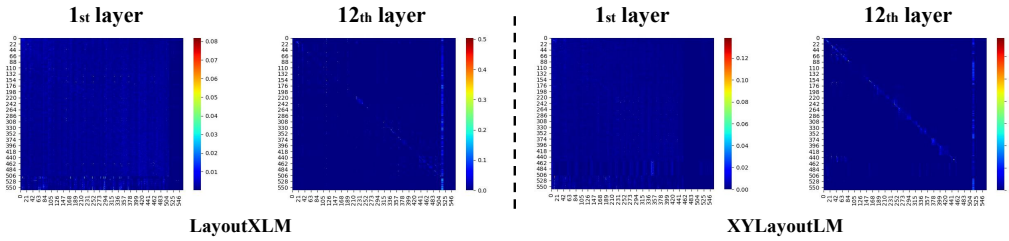
Figure 3. Visualizations of the attention scores from one sample based on LayoutXLM and XYLayoutLM. The attention score maps are from the twelveth attention head in the first/twelveth attention layer. Best viewed in color.
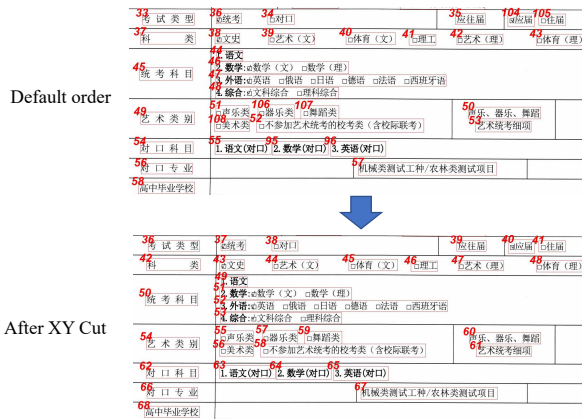

Figure 4. Visualizations of Augmented XY Cut on input tokens. The reading order is shown as the red indices.
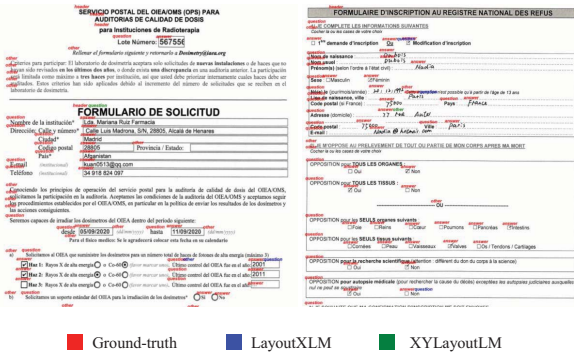

Figure 5. Predictions of LayoutXLM and XYLayoutLM for SER task. The red words are the ground truth. The blue and green words are the prediction of LayoutXLM and XYLayoutLM, respectively. Best viewed in color.

these attention score maps are all from the twelfth attention head without normalization.

From Figure 3 we can draw the following conclusions. The attention weights of XYLayoutLM are larger than LayoutXLM in most layers, which means XYLayoutLM can better capture the attention and relations among tokens. Moreover, benefiting from DCPE, XYLayoutLM can extract more layout information from the local neighbors since the bright lines in XYLayoutLM are bolder.

Augmented XY Cut. We visualize the tokens reading order before and after our proposed Augmented XY Cut in Figure 4. The figure shows that our XY Cut successfully sorts the input tokens in proper reading order.

Performances on XFUN. The visualization of XYLayoutLM and LayoutXLM on the XFUN dataset is shown in Figure 5. The red color in this figure denotes the ground truth, while the blue and green colors mean the predicted categories of LayoutXLM and XYLayoutLM, respectively. The figure shows that our XYLayoutLM can classify tokens better in challenging situations than LayoutXLM.

5. Conclusion

In this work, we introduced XYLayoutLM, a simple yet effective multimodal network for document understanding. Our model contains two related contributions, *i.e.*, Augmented XY Cut for proper reading order and DCPE for generating various-length position embeddings with local layout information. Moreover, it achieves competitive results on several VRDU datasets. We hope our work could inspire designing new frameworks to tackle the challenging document understanding tasks.

6. Acknowledgements

References

[1] M. Aiello and Amw Smeulders. Bidimensional relations for reading order detection. 2003. 3

[2] Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha. Docformer: End-to-end transformer for document understanding. *arXiv preprint arXiv:2106.11539*, 2021. 1, 6, 7

[3] M. Ceci, M. Berardi, G. A. Porcelli, and D. Malerba. A data mining approach to reading order detection. In *ICDAR*, 2007. 3

[4] Zewen Chi, Li Dong, Furu Wei, Nan Yang, Saksham Singhal, Wenhui Wang, Xia Song, Xian-Ling Mao, Heyan Huang, and Ming Zhou. Infoxlm: An information-theoretic framework for cross-lingual language model pre-training. In *NAACL*, 2021. 1, 2, 6

[5] Xiangxiang Chu, Zhi Tian, Bo Zhang, Xinlong Wang, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Conditional positional encodings for vision transformers. *arXiv preprint arXiv:2102.10882*, 2021. 1, 2, 3, 5, 7

[6] Alexis Conneau, Kartikay Khandelwal, Naman Goyal, Vishrav Chaudhary, Guillaume Wenzek, Francisco Guzmán, Edouard Grave, Myle Ott, Luke Zettlemoyer, and Veselin Stoyanov. Unsupervised cross-lingual representation learning at scale. In *ACL*, 2020. 1, 6

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, 2019. 1, 2, 6

[8] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 1, 2

[9] Stefano Ferilli, Domenico Grieco, Domenico Redavid, and Floriana Esposito. Abstract argumentation for reading order detection. In *ACM symposium on Document engineering*, 2014. 3

[10] Łukasz Garncarek, Rafał Powalski, Tomasz Stanisławek, Bartosz Topolski, Piotr Halama, Michał Turski, and Filip Graliński. Lambert: Layout-aware language modeling for information extraction. In *ICDAR*, 2020. 1, 2

[11] Jaekyu Ha, Robert M Haralick, and Ihsin T Phillips. Recursive xy cut using bounding boxes of connected components. In *ICDAE*, 1995. 2, 3, 4

[12] Leipeng Hao, Liangcai Gao, Xiaohan Yi, and Zhi Tang. A table detection method for pdf documents based on convolutional neural networks. In *DAS*, 2016. 1

[13] Teakgyu Hong, Donghyun Kim, Mingi Ji, Wonseok Hwang, Daehyun Nam, and Sungrae Park. Bros: A pre-trained language model focusing on text and layout for better key information extraction from documents. *arXiv preprint arXiv:2108.04539*, 2021. 6

[14] Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. Funsd: A dataset for form understanding in noisy scanned documents. In *ICDARW*, 2019. 5

[15] Anoop Raveendra Katti, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul. Chargrid: Towards understanding 2d documents. In *EMNLP*, 2018. 1

[16] Chenliang Li, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. Structurallm: Structural pre-training for form understanding. *arXiv preprint arXiv:2105.11210*, 2021. 1, 6, 7

[17] Liangcheng Li, Feiyu Gao, Jiajun Bu, Yongpan Wang, Zhi Yu, and Qi Zheng. An end-to-end ocr text re-organization sequence learning for rich-text detail image comprehension. In *ECCV*, 2020. 3

[18] Peizhao Li, Jiuxiang Gu, Jason Kuen, Vlad I Morariu, Handong Zhao, Rajiv Jain, Varun Manjunatha, and Hongfu Liu. Selfdoc: Self-supervised document representation learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5652–5660, 2021. 1, 6, 7

[19] Xiaojing Liu, Feiyu Gao, Qiong Zhang, and Huasha Zhao. Graph convolution for multimodal information extraction from visually rich documents. In *NAACL*, 2019. 1

[20] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. *arXiv preprint arXiv:2103.14030*, 2021. 1, 2

[21] Donato Malerba and Michelangelo Ceci. Learning to order: A relational approach. In *IWMCD*, 2007. 3

[22] Donato Malerba, Michelangelo Ceci, and Margherita Berardi. Machine learning for reading order detection in document image understanding. In *MLDAR*, 2008. 3

[23] George Nagy and Sharad C Seth. Hierarchical representation of optically scanned documents. In *ICPR*, 1984. 4

[24] Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. Deepdesrt: Deep learning for detection and structure recognition of tables in document images. In *ICDAR*, 2017. 1

[25] Peter Shaw, Jakob Uszkoreit, and Ashish Vaswani. Self-attention with relative position representations. In *NAACL-HLT*, 2018. 2

[26] Carlos Soto and Shinjae Yoo. Visual detection with context for document layout analysis. In *EMNLP*, 2019. 1

[27] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, 2017. 1, 2

[28] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. Layoutlm: Pre-training of text and layout for document image understanding. In *KDD*, 2020. 1, 2, 6

[29] Yiheng Xu, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, and Furu Wei. Layoutxlm: Multimodal pre-training for multilingual visually-rich document understanding. *arXiv preprint arXiv:2104.08836*, 2021. 1, 2, 3, 5, 6

[30] Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. In *ACL*, 2021. 1, 2, 3, 6

[31] Lecun Yann, Bottou Leon, Bengio Yoshua, and Haffner Patrick. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 1

[32] Fisher Yu and Vladlen Koltun. Multi-scale context aggregation by dilated convolutions. *arXiv preprint arXiv:1511.07122*, 2015. 5

[33] Z. Yu, J. Yu, Y. Cui, D. Tao, and Q. Tian. Deep modular co-attention networks for visual question answering. In *CVPR*, 2019. 1

[34] Xu Zhong, Jianbin Tang, and Antonio Jimeno Yepes. Publaynet: largest dataset ever for document layout analysis. In *ICDAR*, 2019. 1

[35] Wang Zilong, Xu Yiheng, Cui Lei, Shang Jingbo, and Wei Furu. Layoutreader: Pre-training of text and layout for reading order detection. In *EMNLP*, 2021. 3