BROS: A PRE-TRAINED LANGUAGE MODEL FOR UNDERSTANDING TEXTS IN DOCUMENT

Anonymous authors
Paper under double-blind review

ABSTRACT

Understanding document from their visual snapshots is an emerging and challenging problem that requires both advanced computer vision and NLP methods. Although the recent advance in OCR enables the accurate extraction of text segments, it is still challenging to extract key information from documents due to the diversity of layouts. To compensate for the difficulties, this paper introduces a pre-trained language model, *BERT Relying On Spatiality (BROS)*, that represents and understands the semantics of spatially distributed texts. Different from previous pre-training methods on 1D text, BROS is pre-trained on large-scale semi-structured documents with a novel area-masking strategy while efficiently including the spatial layout information of input documents. Also, to generate structured outputs in various document understanding tasks, BROS utilizes a powerful graph-based decoder that can capture the relation between text segments. BROS achieves state-of-the-art results on four benchmark tasks: FUNSD, SROIE*, CORD, and SciTSR. *Our experimental settings and implementation codes will be publicly available*.

1 INTRODUCTION

Document intelligence (DI)[1], which understands industrial documents from their visual appearance, is a critical application of AI in business. One of the important challenges of DI is a key information extraction task (KIE) (Huang et al., 2019; Jaume et al., 2019; Park et al., 2019) that extracts structured information from documents such as financial reports, invoices, business emails, insurance quotes, and many others. KIE requires a multi-disciplinary perspective spanning from computer vision for extracting text from document images to natural language processing for parsing key information from the identified texts.

Optical character recognition (OCR) is a key component to extract all texts in document images. As OCR provides a set of text blocks consisting of a text and its location, key information in documents can be represented as a single or a sequence of the text blocks (Schuster et al., 2013; Qian et al., 2019; Hwang et al., 2019; 2020). Although OCR alleviates the burden of processing images, understanding semantic relations between text blocks on diverse layouts remains a challenging problem.

To solve this problem, existing works use a pre-trained language model to utilize its effective representation of text. Hwang et al. (2019) fine-tunes BERT by regarding KIE tasks as sequence tagging problems. Denk & Reisswig (2019) uses BERT to incorporate textual information into image pixels during their image segmentation tasks. However, since BERT is designed for text sequences, they artificially convert text blocks distributed in two dimensions into a single text sequence losing spatial layout information. Recently, Xu et al. (2020) proposes LayoutLM pre-trained on large-scale documents by utilizing spatial information of text blocks. They show the effectiveness of the pre-training approach by achieving high performance on several downstream tasks. Despite this success, LayoutLM has three limitations. First, LayoutLM embeds x- and y-axis individually using trainable parameters like the position embedding of BERT, ignoring the gap between positions in a sequence and 2D space. Second, its pre-training method is essentially identical to BERT that does not explicitly consider spatial relations between text blocks. Finally, in its downstream tasks, LayoutLM only conducts sequential tagging tasks (e.g. BIO tagging) that require serialization of text blocks.

[1] https://sites.google.com/view/di2019

These limitations indicate that LayoutLM fails not only to fully utilize spatial information but also to address KIE problems in practical scenarios when a serialization of text blocks is difficult.

This paper introduces an advanced language model, BROS, pre-trained on large-scale documents, and provides a new guideline using the model to solve KIE tasks. Specifically, to address the three limitations mentioned above, BROS combines three proposed methods: (1) a 2D positional encoding method that can represent the continuous property of 2D space, (2) a novel area-masking pre-training strategy that performs masked language modeling on 2D, and (3) a combination with a graph-based decoder for solving KIE tasks. We evaluated BROS on four public KIE datasets: FUNSD (form-like documents), SROIE* (receipts), CORD (receipts), and SciTSR (table structures) and observed that BROS achieved the best results on all tasks. In addition, to reflect real KIE problems with no order of text blocks, we removed order information of the four benchmark datasets. When comparing models on these modified datasets, BROS still shows the best performance. Further ablation studies provide how each component contributes to the final performances of BROS.

2 Related Work

2.1 Pre-trained Language Models

BERT (Devlin et al., 2019) is a pre-trained language model using Transformer (Vaswani et al., 2017) that shows superior performance on various NLP tasks. The main strategy to train BERT is a masked language model (MLM) that masks and estimates randomly selected tokens to learn the semantics of language from large-scale corpora. Many variants of BERT have been introduced to learn transferable knowledge by modifying the pre-training strategy. XLNet (Yang et al., 2019) permutes tokens during the pre-training phase to reduce a discrepancy from the fine-tuning phase and utilizes relative position encoding to handle long texts. StructBERT (Wang et al., 2020) shuffles tokens in text spans and adds sentence prediction tasks for recovering the order of words or sentences. SpanBERT (Joshi et al., 2020) masks span of tokens to extract better representation for span selection tasks such as question answering and co-reference resolution. ELECTRA (Clark et al., 2020) is trained to distinguish real and fake input tokens generated by another network for sample-efficient pre-training.

Inspired by these previous works, BROS utilizes a new pre-training strategy that can capture complex spatial dependencies between text blocks distributed on two dimensions. Note that LayoutLM is the first pre-trained language model on spatial text blocks but they still use the original MLM of BERT.

2.2 Key Information Extraction from Documents

Most of the existing approaches utilize a serializer to identify the text order of key information. POT (Hwang et al., 2019) applies BERT on serialized text blocks and extracts key contexts via a BIO tagging approach. CharGrid (Katti et al., 2018) and BERTGrid (Denk & Reisswig, 2019) map text blocks upon a grid space, identify the region of key information, and extract key contexts in the pre-determined order. Liu et al. (2019), Yu et al. (2020), and Qian et al. (2019) utilize graph convolutional networks to model dependencies between text blocks but their decoder that performs BIO tagging relies on a serialization. LayoutLM (Xu et al., 2020) is pre-trained on large-scale documents with spatial information of text blocks, but it also conducts BIO tagging for their downstream tasks. However, using a serializer and relying on the identified sequence has two limitations. One is that the original meaning can be lost by improper serialization and the other is that there may even be no correct serialization order.

A natural way to model key contexts from text blocks is a graph-based formulation that identifies all relationships between text blocks. SPADE (Hwang et al., 2020) proposes a graph-based decoder to extract key contexts from identified connectivity between text blocks without any serialization. Specifically, they utilize BERT without its sequential position embeddings and train the model while fine-tuning BERT. However, their performance is limited by the amount of data as all relations have to be learned from the beginning at the fine-tuning stage. To fully utilize the graph-based decoder, BROS is pre-trained on a large number of documents and is combined with the SPADE decoder to determine key contexts from text blocks.

3 BERT RELYING ON SPATIALITY (BROS)

Since BROS is an extension of BERT, the main structure of BROS follows BERT, but there are three novel differences: (1) a spatial encoding metric that reflects the continuous property of 2D space, (2) a pre-training objective designed for text blocks on 2D space, and (3) a guideline for designing downstream models based on a graph-based formulation.

3.1 ENCODING SPATIAL INFORMATION INTO BERT

3.1.1 REPRESENTATION OF A TEXT BLOCK LOCATION

The way to represent spatial information of text blocks is important to reveal spatial dependencies between text blocks. We utilize sinusoidal functions to encode continuous values of x- and y-axis, and merge them through a linear transformation to represent a point upon 2D space.

For formal description, let denote $\boldsymbol{p} = (x, y)$ as a point on 2D space and $\mathbf{f}^{\text{sinu}} : \mathbb{R} \rightarrow \mathbb{R}^{D^{\text{s}}}$ as a sinusoidal function. D^{s} is the dimensions of sinusoid embedding. BROS encodes a 2D point by applying the sinusoidal function to x- and y-axis and concatenating them as $\bar{\boldsymbol{p}} = [\mathbf{f}^{\text{sinu}}(x) \oplus \mathbf{f}^{\text{sinu}}(y)]$. The $\oplus$ symbol indicates concatenation. Since location of a text block, bb_i, forms as four vertices, $\boldsymbol{p}_i^{\text{tl}}$, $\boldsymbol{p}_i^{\text{tr}}$, $\boldsymbol{p}_i^{\text{br}}$ and $\boldsymbol{p}_i^{\text{bl}}$, of its bounding box, all points are converted into vectors as $\bar{\boldsymbol{p}}_i^{\text{tl}}$, $\bar{\boldsymbol{p}}_i^{\text{tr}}$, $\bar{\boldsymbol{p}}_i^{\text{br}}$, and $\bar{\boldsymbol{p}}_i^{\text{bl}}$. Finally, to represent a spatial embedding, $\overline{\boldsymbol{bb}}_i$, of a text block, BROS combines four identified vectors through a linear transformation,

$$\overline{\boldsymbol{bb}}_i = \boldsymbol{W}^{\text{tl}}\bar{\boldsymbol{p}}_i^{\text{tl}} + \boldsymbol{W}^{\text{tr}}\bar{\boldsymbol{p}}_i^{\text{tr}} + \boldsymbol{W}^{\text{br}}\bar{\boldsymbol{p}}_i^{\text{br}} + \boldsymbol{W}^{\text{bl}}\bar{\boldsymbol{p}}_i^{\text{bl}}, \tag{1}$$

where $\boldsymbol{W}^{\text{tl}}, \boldsymbol{W}^{\text{tr}}, \boldsymbol{W}^{\text{br}}, \boldsymbol{W}^{\text{bl}} \in \mathbb{R}^{H \times 2D^{\text{s}}}$ are linear transition metrics and H is a hidden size of BERT. Since the sinusoidal function has a periodic property, it is more natural to represent continuous 2D point rather than using point-specific parameters like BERT. In addition, by learning the linear transition parameters, BROS provides an effective representation of a bounding box.

3.1.2 ENCODING SPATIAL REPRESENTATION

A way to encode the position embeddings to the model decides how to utilize the position information. In BERT, position embedding is tied with the token through a point-wise summation. However, 2D spatial information is richer than 1D sequence due to the continuous property and the high dimensionality. Moreover, a text block can exist over all locations in the document without any changes in its semantic meaning. For example, locations of page numbers differ over multiple document snapshots even though they are captured for a single document. Therefore, spatial information should be encoded more carefully than simply summing on the token embeddings.

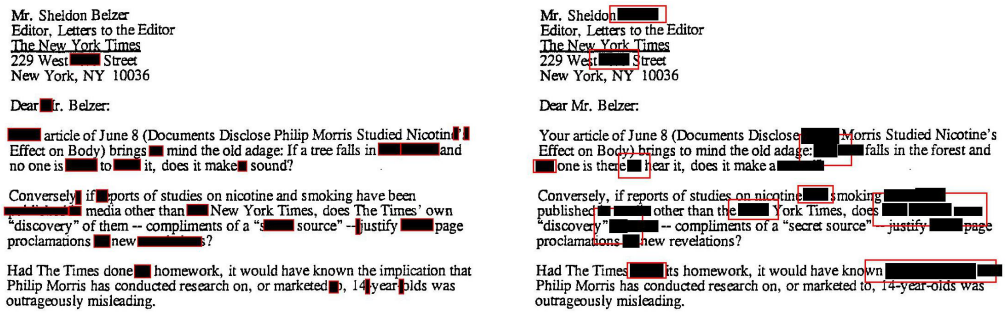
Instead of using the traditional position encoding method, BROS encodes spatial information directly when identifying relationships between text blocks. Specifically, BROS calculates an attention logit combining semantic and spatial comparison. The former is the same as the original attention mechanism in Transformer (Vaswani et al., 2017), but the latter is a new method identifying the importance of the target location when the source context and location are given. Our proposed attention logit is formulated as follows,

$$A_{i,j} = (\boldsymbol{W}^{\text{q}}\boldsymbol{t}_i)^\top(\boldsymbol{W}^{\text{k}}\boldsymbol{t}_j) + (\boldsymbol{W}^{\text{q}}\boldsymbol{t}_i \odot \boldsymbol{W}^{\text{sq|q}}\overline{\boldsymbol{bb}}_i)^\top(\boldsymbol{W}^{\text{sk|q}}\overline{\boldsymbol{bb}}_j) + (\boldsymbol{W}^{\text{sq}}\overline{\boldsymbol{bb}}_i)^\top(\boldsymbol{W}^{\text{sk}}\overline{\boldsymbol{bb}}_j), \tag{2}$$

where $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$ are context representations for i^{th} and j^{th} tokens and $\boldsymbol{W}^{\text{q}}$, $\boldsymbol{W}^{\text{k}}$, $\boldsymbol{W}^{\text{sq|q}}$, $\boldsymbol{W}^{\text{sk|q}}$, $\boldsymbol{W}^{\text{sq}}$, $\boldsymbol{W}^{\text{sk}}$ are linear transition matrices. The $\odot$ symbol indicates Hadamard product. The first term indicates an attention logit from contextual representations and the third term is from spatial representation. The second term is designed to model the spatial dependency given the source semantic representation, $\boldsymbol{t}_i$. The second and third terms are independently calculated at each layer because spatial dependencies might differ over layers.

3.2 PRE-TRAINING OBJECTIVE: AREA-MASKED LANGUAGE MODEL

Pre-training diverse layouts from unlabeled documents is a key factor for document understanding tasks. To learn effective spatial representation including relationships between text blocks, we propose a novel pre-training objective for learning a language with spatial layouts. Inspired by



(a) Random *token* selection and masking

(b) Random *area* selection and masking

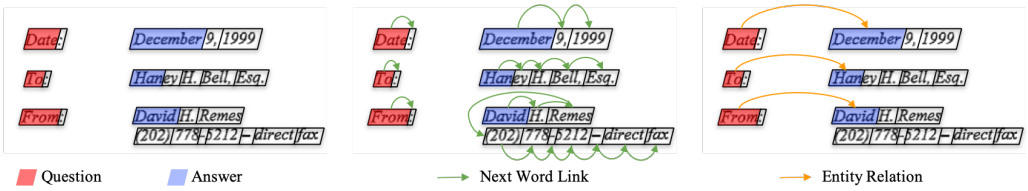
Figure 1: Examples of text masking methods to learn masked language models. Red boxes indicate random selections and black boxes show the masked text. In both, 15% of tokens are masked.

SpanBERT (Joshi et al., 2020), we expand spans of a 1D sequence to consecutive text blocks in 2D space. Specifically, we select a few regions in 2D space, mask tokens in the selected regions, and estimate the tokens. The rule of masking an area follows below stages: (1) randomly selects a text block, (2) adjusts the region of text block by multiplying a number sampled from an exponential distribution with a hyper-parameter, λ, (3) masks all tokens in text blocks of which center position is in the adjusted area. The rationale behind using exponential distribution is to convert the geometric distribution used in SpanBERT for a discrete domain into distribution for a continuous domain. In detail, we truncated exponential distribution with 1 to prevent an infinity multiplier covering all space of the document. This area-masking is repeatedly conducted until 15% of tokens are masked. Figure 1 compares token- and area-masking on text blocks.

3.3 Spatial Dependency Parsers for Downstream Tasks

Key information in a document (e.g. store address in a receipt) is represented as sub-sequences of text blocks. Although BIO tagging is a general approach to extract sub-sequences in a text sequence, it cannot represent key texts in a document without the optimal order of text blocks. Therefore, BIO tagging cannot be applied in a practical scenario because it is impossible to access the optimal order.

To deal with the issue, BROS utilizes a decoder of SPADE (Hwang et al., 2020) that can infer a sequence of text blocks from a graph-based formulation without any order information. BROS supports two downstream tasks. One is an entity extraction (EE) task and the other is an entity linking (EL) task. The EE identifies a sequence of text blocks for key information (e.g. extract address texts in a receipt) and the EL determines relations between target texts when target text blocks are known (e.g. identify key and value text pairs).



(a) Starting token classification

(b) Next token classification

(c) Entity linking (EL) task

Figure 2: Visual descriptions of BROS downstream tasks. For EE tasks, BROS combines two sub-tasks such as (a) and (b). For EL task, BROS links the first tokens of the entities.

For EE tasks, BROS divides the problem into two sub-tasks, starting token classification (Figure 2, a) and next token classification (Figure 2, b). Let $\tilde{\boldsymbol{t}}_i \in \mathbb{R}^H$ as the ith token representation from the last Transformer layer of BROS. The starting token classification conducts a token-level tagging determining whether a token is a starting token of target information as follows,

$$\mathbf{f}_{\text{stc}}(\tilde{\boldsymbol{t}}_i) = \text{softmax}(\boldsymbol{W}^{\text{stc}}\tilde{\boldsymbol{t}}_i), \tag{3}$$

where $\boldsymbol{W}^{\text{stc}} \in \mathbb{R}^{(C+1)\times H}$ is a linear transition matrix and C indicates the number of target classes. Here, the output dimension has one more dimension than C to classify the tokens which are not starting tokens. The next token classification is conducted by utilizing pair-wise token representations as follows,

$$\mathbf{f}_{\text{ntc}}(\tilde{\boldsymbol{t}}_i) = \text{softmax}((\boldsymbol{W}^{\text{ntc-s}}\tilde{\boldsymbol{t}}_i)^\top(\boldsymbol{t}^{ntc} \oplus \boldsymbol{W}^{\text{ntc-t}}\tilde{\boldsymbol{t}}_1 \oplus \cdots \oplus \boldsymbol{W}^{\text{ntc-t}}\tilde{\boldsymbol{t}}_N)), \tag{4}$$

where $\boldsymbol{W}^{\text{ntc-s}}, \boldsymbol{W}^{\text{ntc-t}} \in \mathbb{R}^{H^{\text{ntc}}\times H}$ are linear transition matrices, H^{ntc} is a hidden feature dimension for the next token classification decoder and N is the maximum number of tokens. Here, $\boldsymbol{t}^{\text{ntc}} \in \mathbb{R}^{H^{\text{ntc}}}$ is a model parameter to classify tokens which are not included in target classes. By solving two subtasks, BROS can identify a sequence of text blocks by finding first tokens and connecting next tokens for each class.

For EL tasks, BROS conducts a binary classification for all possible pairs of tokens (Figure 2, c) as follows,

$$\mathbf{f}_{\text{rel}}(\tilde{\boldsymbol{t}}_i, \tilde{\boldsymbol{t}}_j) = \text{sigmoid}((\boldsymbol{W}^{\text{rel-s}}\tilde{\boldsymbol{t}}_i)^\top(\boldsymbol{W}^{\text{rel-t}}\tilde{\boldsymbol{t}}_j)), \tag{5}$$

where $\boldsymbol{W}^{\text{rel-s}}, \boldsymbol{W}^{\text{rel-t}} \in \mathbb{R}^{H^{\text{rel}}\times H}$ are linear transition matrices and H^{rel} is a hidden feature dimension. The difference from the next token classification is that a single token can hold multiple relations with other tokens to represent hierarchical structures of document layouts. For more detail about this graph-based formulation, please refer to Hwang et al. (2020).

4 Key Information Extraction Tasks

Here, we describe three EE tasks and three EL tasks from four benchmark datasets for KIE.

- Form Understanding in Noisy Scanned Documents (FUNSD) (Jaume et al., 2019) is a set of documents with various forms. The dataset consists of 149 training and 50 testing examples. FUNSD has both EE and EL tasks. In the EE task, there are three semantic entities: Header, Question, and Answer. In the EL task, the semantic hierarchies are represented as relations between text blocks like header-question and question-answer pairs.
- SROIE* is a variant of Task 3 of "Scanned Receipts OCR and Information Extraction" (SROIE)[2] that consists of a set of store receipts. In the original SROIE task, semantic contents (Company, Date, Address, and Total price) are generated without explicit connection to the text blocks. To convert SROIE into a EE task, we developed SROIE* by matching ground truth contents with text blocks. We also split the original training set into 526 training and 100 testing examples because the ground truths are not given in the original test set. SROIE* will be publicly available.
- Consolidated Receipt Dataset (CORD) (Park et al., 2019) is a set of store receipts with 800 training, 100 validation, and 100 testing examples. CORD consists of both EE and EL tasks. In the EE task, there are 30 semantic entities including menu name, menu price, and so on. In the EL task, the semantic entities are linked according to their layout structure. For example, menu name entities are linked to menu id, menu count, and menu price.
- Complicated Table Structure Recognition (SciTSR) (Chi et al., 2019) is a EL task that connects cells in a table to recognize the table structure. There are two types of relations: vertical and horizontal connection between cells. The dataset consists of 12,000 training images and 3,000 test images.

Although, these four datasets provide test beds for the EE and EL tasks, they represent the subset of real problems as the optimal order of text blocks is given. In real service, user can submit documents with a complex layout where the serialization of input texts are non-trivial. FUNSD provides the optimal orders of text blocks related to target classes in both training and testing examples. In SROIE*, CORD, and SciTSR, the text blocks are serialized in reading orders.

To reveal the serialization problem in the EE and EL tasks, we randomly permuted text blocks of the datasets to remove their order information. We denote the permuted datasets as p-FUNSD, p-SROIE*, p-CORD, and p-SciTSR and compare all models on them. For fair comparisons, we will open the permuted datasets.

[2]https://rrc.cvc.uab.es/?ch=13

Dataset	Model	Entity Extraction			Entity Linking		
		P	R	F1	P	R	F1
FUNSD	BERT (Xu et al., 2020)	54.69	61.70	60.26	5.22	0.56	0.98
	LayoutLM (Xu et al., 2020)	75.97	81.55	78.66	-	-	-
	LayoutLM†	76.12	81.88	78.89	41.29	44.45	42.81
	BROS	**80.56**	**81.88**	**81.21**	**64.30**	**69.86**	**66.96**
SROIE*	BERT	92.90	94.47	93.67			
	LayoutLM†	94.31	95.78	95.04		-	
	BROS	**94.93**	**96.03**	**95.48**			
CORD	BERT	93.08	93.18	93.13	58.40	42.00	48.08
	LayoutLM†	95.03	94.58	94.80	91.37	89.52	90.43
	BROS	**95.58**	**95.14**	**95.36**	**92.24**	**90.89**	**91.56**
SciTSR	Tabby (Shigarov et al., 2016)				92.6	92.0	92.1
	DeepDeSRT (Schreiber et al., 2017)				90.6	88.7	89.0
	GraphTSR (Chi et al., 2019)		-		95.9	94.8	95.3
	BERT				87.61	85.92	86.76
	LayoutLM†				98.76	99.44	99.09
	BROS				**99.05**	**99.47**	**99.26**

Table 1: Performance comparisons on three EE and three EL tasks with the *optimal order information* of spatial text blocks. P, R, and F1 indicate precision, recall, and F1 score, respectively. For EL tasks connecting entities, SPADE decoder is applied for all pre-trained models such as BERT, LayoutLM, and BROS. The scores are the average of the results with 5 different random seeds.

5 EXPERIMENTS

5.1 EXPERIMENT SETTINGS

For pre-training, IIT-CDIP Test Collection 1.0[3] (Lewis et al., 2006), which consists of approximatley 11M document images, is used but 400K RVL-CDIP dataset[4] (Harley et al., 2015) is excluded following LayoutLM. In-house OCR engine was applied to obtain text blocks from unlabeled document images.

The main Transformer structure of BROS is the same as BERT. By following $\text{BERT}_{\text{BASE}}$, the hidden size, the number of self-attention heads, the feed-forward size, and the number of Transformer layers are 768, 12, 3072, and 12, respectively. Our pre-training setting is similar to LayoutLM for a fair comparison. BROS is trained by using AdamW optimizer (Loshchilov & Hutter, 2019) with a learning rate of 5e-5 with linear decay. First 10% of the total epochs are used for a warm-up. We initialized weights of BROS with those of $\text{BERT}_{\text{BASE}}$ and trained BROS on IIT-CDIP for 2 epochs with 80 of batch size, following LayoutLM. The pre-training takes 64 hours with 8 NVIDIA Tesla V100s with Distributed Data Parallel (DDP).

During fine-tuning, the learning rate is set to 5e-5. The batch size is set to 16 for all tasks. The number of training epochs or steps is as follows: 100 epochs for FUNSD, 1K steps for SROIE* and CORD, and 7.5 epochs for SciTSR. The hidden feature dimensions, H^{ntc} and H^{rel}, of the SPADE decoder are set to 128 for FUNSD, 64 for SROIE*, and 256 for CORD and SciTSR.

Although the authors of LayoutLM published their codes on GitHub[5], the data and script file used for pre-training are not included. For a fair comparison, we made our own implementation, which we refer to LayoutLM†, on the same pre-training data and script file used for BROS pre-training. We verified LayoutLM† by comparing its performances on FUNSD from the reported scores in (Xu et al., 2020). See Appendix A for more information.

[3]https://ir.nist.gov/cdip/

[4]https://www.cs.cmu.edu/ aharley/rvl-cdip/

[5]https://github.com/microsoft/unilm/tree/master/layoutlm

5.2 EXPERIMENTAL RESULTS WITH OPTIMAL ORDER INFORMATION

Table 1 shows the results on four KIE datasets with given optimal order information of text blocks. For EL tasks, we applied SPADE decoders to all pre-trained models such as BERT, LayoutLM, and BROS. In all tasks, we observed that BERT shows lower scores than LayoutLM and BROS presumably due to the loss of spatial information. BROS achieved the highest scores showing effectiveness of our approach. Specifically, in FUNSD, BROS shows the state-of-the-art performances with a large margins of 2.32pp in the EE task and 24.15pp in the EL task. Moreover, it should be noted that BROS achieves higher f1 score than 79.27 of LayoutLM using visual features. In SROIE* and CORD, BROS also shows the best performances over all the EE and EL tasks. In SciTSR, LayoutLM and BROS show the importance of pre-training by exceeding other baselines with large margins which are trained by using either only spatial information of cells (Tabby and DeepDeSRT) or without pre-training spatial texts (GraphTSR). By achieving the best, these results prove that BROS captures better representations of text blocks for KIE downstream tasks.

5.3 EXPERIMENTAL RESULTS WITHOUT OPTIMAL ORDER INFORMATION

It is an another challenging problem to arrange text blocks in the order that humans understand it (Li et al., 2020). Since the experiments in Section 5.2 assumes the optimal order of text blocks, they cannot reflect real KIE problems. To reveal the challenge, we removed the order information in all datasets by permuting the order of text blocks as mentioned in Section 4 and investigated how BERT, LayoutLM, and BROS work without the order information. We utilized a SPADE decoder for all models because BIO tagging on these permuted dataset cannot extract a sequence of text blocks in a correct order.

		Entity Extraction			Entity Linking		
Dataset	Model	P	R	F1	P	R	F1
p-FUNSD	BERT	7.91	4.86	5.87	3.77	0.53	0.89
	LayoutLM†	34.98	32.55	33.72	36.69	31.25	33.75
	BROS	**74.84**	**75.50**	**75.14**	**64.18**	**68.04**	**66.05**
p-SROIE	BERT	7.29	2.56	3.79		-	
	LayoutLM†	67.42	66.48	66.94			
	BROS	**81.32**	**81.36**	**81.34**			
p-CORD	BERT	20.72	19.56	20.12	37.69	4.99	8.32
	LayoutLM†	77.49	77.27	77.38	58.26	56.38	57.30
	BROS	**93.86**	**93.68**	**93.77**	**84.56**	**83.03**	**83.79**
p-SciTSR	BERT		-		59.42	0.89	1.75
	LayoutLM†				95.59	99.04	97.28
	BROS				**98.90**	**99.32**	**99.11**

Table 2: Performance comparisons on three EE and three EL tasks without the optimal order information of text blocks. All model utilize a SPADE decoder because BIO tagging does not work without the optimal order. The scores are the average of the results with 5 different random seeds.

Table 2 shows the results. Due to the lose of correct orders, BERT shows poor performances over all tasks. By utilizing spatial information of text blocks, LayoutLM† shows better performance but it suffers from huge performance degradation compared to the score computed with the optimal order. On the other hand, BROS shows comparable results compared the cases with the optimal order and achieves better performances than BERT and LayoutLM†.

To systematically investigate how the order information affects the performance of the models, we construct variants of FUNSD by re-ordering text blocks with two sorting methods based on the top-left points. The text blocks of xy-FUNSD are sorted according to x-axis with ascending order of y-axis and those of yx-FUNSD are sorted according to y-axis with ascending order of x-axis.

Table 3 shows performance on p-FUNSD, xy-FUNSD, yx-FUNSD, and the original FUNSD. All models utilize a SPADE decoder for a fair comparison. Interestingly, the performance of LayoutLM† is degraded in the order of FUNSD, yx-FUNSD, xy-FUNSD, and p-FUNSD as like the order of the

Model	p-FUNSD	xy-FUNSD	yx-FUNSD	FUNSD
LayoutLM†	33.72	36.61	60.50	77.30
BROS	75.14	75.24	76.26	81.21

Table 3: Comparison of the EE task performance changes according to sorting methods. p-FUNSD and FUNSD indicate permuted and original datasets, repectively. xy-FUNSD/yx-FUNSD represents datasets which text blocks are sorted on x/y-axis with the top-left point of the blocks and then sorted on y/x-axis again for holding same values. The scores indicate averages of F1 scores for the EE task and they are repeated five times.

reasonable serialization for text on 2D space. On the other hand, the performance of BROS is relatively consistent. These results show that BROS is applicable to real KIE problems without additional serialization method.

5.4 Ablation Studies

MLM	**F1**
Random tokens (LayoutLM's)	73.85
Random areas	74.44
2D position encoding method	**F1**
Tied with token embedding (LayoutLM's)	42.46
Untied with token embedding	74.44

2D position embedding method	**F1**
Look-up tables (LayoutLM's)	67.99
Sinusoid & linear	74.44
Decoder for downstream tasks	**F1**
BIOE tagging (LayoutLM's)	73.16
SPADE decoder	74.44

Table 4: Ablation studies of the proposed changes from LayoutLM's on the FUNSD EE task. The number of pre-training data is 512K. The scores are the average of 5 experimental results.

Table 4 provides ablative experiments by changing MLM of pre-training strategy, 2D position embedding and encoding methods, and a decoder for downstream tasks. The 2D embedding method represents how to treat spatial information of text blocks and the 2D encoding method indicates how to merge the 2D embeddings into BERT. The results show that all modifications improve the performance when comparing the methods of LayoutLM. Specifically, 2D position embedding and its encoding methods show huge performance gaps by 6.45pp and 31.98pp, respectively. These results represent a co-modality of our 2D continuous position embedding approach and its untied encoding method.

LayoutLM and BROS are initialized with weights of BERT to utilize powerful knowledge of BERT that learns from large-scale corpora. However, BERT includes its 1D positional embeddings (1D-PE) that might be harmful by making a sequence of text blocks even though there is no order information. To investigate the effectiveness of the 1D-PE, we conduct an ablative study. BROS without the 1D-PE shows the same F1 scores on both FUNSD and p-FUNSD (70.07), but BROS with the 1D-PE shows performance degradation when the dataset loses the optimal order information (81.21 on FUNSD $\rightarrow$ 75.14 on p-FUNSD). Nevertheless, BROS with the 1D-PE shows better performances on both datasets. This might be because the 1D-PE preserves the token order in a single text block. Based on this result, we decided to incorporate the 1D-PE in our model.

6 Conclusion

We present BROS that encodes 2D continuous position of text blocks as well as learns natural language from text blocks with an area-driven training strategy. To extract key contexts from text blocks without the order information, BROS adapts a graph-based decoder, SPADE. In our extensive experiments on four KIE tasks, BROS consistently shows the best performances as well as the robustness on the order of text blocks compared to the existing approaches.

REFERENCES

Zewen Chi, Heyan Huang, Heng-Da Xu, Houjin Yu, Wanxuan Yin, and Xian-Ling Mao. Complicated table structure recognition. *arXiv preprint arXiv:1908.04729*, 2019.

Kevin Clark, Minh-Thang Luong, Quoc V Le, and Christopher D Manning. ELECTRA: Pre-training text encoders as discriminators rather than generators. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020.

Timo I Denk and Christian Reisswig. BERTgrid: Contextualized embedding for 2d document representation and understanding. In *Workshop on Document Intelligence at NeurIPS 2019*, 2019.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 1 (Long and Short Papers)*, pp. 4171–4186, 2019.

Adam W. Harley, Alex Ufkes, and Konstantinos G. Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 991–995, 2015.

Zheng Huang, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and CV Jawahar. ICDAR2019 competition on scanned receipt ocr and information extraction. In *Proceedings of the 15th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1516–1520. IEEE, 2019.

Wonseok Hwang, Seonghyeon Kim, Minjoon Seo, Jinyeong Yim, Seunghyun Park, Sungrae Park, Junyeop Lee, Bado Lee, and Hwalsuk Lee. Post-OCR parsing: building simple and robust parser via bio tagging. In *Workshop on Document Intelligence at NeurIPS 2019*, 2019.

Wonseok Hwang, Jinyeong Yim, Seunghyun Park, Sohee Yang, and Minjoon Seo. Spatial dependency parsing for 2d document understanding. *arXiv preprint arXiv:2005.00642*, 2020.

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. FUNSD: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, pp. 1–6. IEEE, 2019.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S Weld, Luke Zettlemoyer, and Omer Levy. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics (TACL)*, 8:64–77, 2020.

Anoop R Katti, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul. Chargrid: Towards understanding 2d documents. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 4459–4469, 2018.

David Lewis, Gady Agam, Shlomo Argamon, Ophir Frieder, D Grossman, and Jefferson Heard. Building a test collection for complex document information processing. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval (SIGIR)*, pp. 665–666, 2006.

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 *Proceedings of the 16th European Conference on Computer Vision (ECCV)*, 2020.

Xiaojing Liu, Feiyu Gao, Qiong Zhang, and Huasha Zhao. Graph convolution for multimodal information extraction from visually rich documents. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 2 (Industry Papers)*, pp. 32–39, 2019.

Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*, 2019.

Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. CORD: A consolidated receipt dataset for post-ocr parsing. In *Workshop on Document Intelligence at NeurIPS 2019*, 2019.

Yujie Qian, Enrico Santus, Zhijing Jin, Jiang Guo, and Regina Barzilay. GraphIE: A graph-based framework for information extraction. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT), Volume 1 (Long and Short Papers)*, pp. 751–761, 2019.

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 *Proceedings of the 14th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 1162–1167. IEEE, 2017.

Daniel Schuster, Klemens Muthmann, Daniel Esser, Alexander Schill, Michael Berger, Christoph Weidling, Kamil Aliyev, and Andreas Hofmeier. Intellix–end-user trained information extraction for document archiving. In *Proceedings of the 12th International Conference on Document Analysis and Recognition (ICDAR)*, pp. 101–105. IEEE, 2013.

Alexey Shigarov, Andrey Mikhailov, and Andrey Altaev. Configurable table structure recognition in untagged PDF documents. In *Proceedings of the 2016 ACM Symposium on Document Engineering (DocEng)*, pp. 119–122, 2016.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, pp. 5998–6008, 2017.

Wei Wang, Bin Bi, Ming Yan, Chen Wu, Jiangnan Xia, Zuyi Bao, Liwei Peng, and Luo Si. StructBERT: Incorporating language structures into pre-training for deep language understanding. In *Proceedings of the 8th International Conference on Learning Representations (ICLR)*, 2020.

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 *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD)*, pp. 1192–1200, 2020.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Ruslan Salakhutdinov, and Quoc V Le. XLNet: Generalized autoregressive pretraining for language understanding. In *Advances in Neural Information Processing Systems 32 (NeurIPS)*, pp. 5753–5763, 2019.

Wenwen Yu, Ning Lu, Xianbiao Qi, Ping Gong, and Rong Xiao. PICK: Processing key information extraction from documents using improved graph learning-convolutional networks. In *Proceedings of the 25th International Conference on Pattern Recognition (ICPR)*, 2020.

A REPRODUCING THE LAYOUTLM

# Pre-training Data	# Epochs	Model	P	R	F1
500K	1	LayoutLM (Xu et al., 2020)	0.5779	0.6955	0.6313
		LayoutLM †	0.5823	0.6935	0.6330
1M	1	LayoutLM (Xu et al., 2020)	0.6156	0.7005	0.6552
		LayoutLM†	0.6142	0.7151	0.6608
2M	1	LayoutLM (Xu et al., 2020)	0.6599	0.7355	0.6957
		LayoutLM†	0.6562	0.7456	0.6980
11M	1	LayoutLM (Xu et al., 2020)	0.7464	0.7815	0.7636
		LayoutLM†	0.7384	0.8022	0.7689
	2	LayoutLM (Xu et al., 2020)	0.7597	0.8155	0.7866
		LayoutLM†	0.7612	0.8188	0.7889

Table 5: Sanity checking of LayoutLM† by comparing its performances on FUNSD from the reported scores in Xu et al. (2020).

As mentioned in the paper, to compare BROS from LayoutLM in diverse experimental settings, we implement LayoutLM in our experimental pipeline. Table 5 compares our implementations from the reported scores in Xu et al. (2020). As can be seen, multiple experiments are conducted according to the number of pre-training data. Our implementation, referred to LayoutLM†, shows comparable performances over all settings.