BROS: A Pre-trained Language Model Focusing on Text and Layout for Better Key Information Extraction from Documents

Teakgyu Hong[1], **Donghyun Kim**[1], **Mingi Ji**[2], **Wonseok Hwang**[3], **Daehyun Nam**[4], **Sungrae Park**[4]
[1]NAVER CLOVA, [2]KAIST, [3]LBox, [4]Upstage AI Research, Upstage AI
teakgyu.hong@navercorp.com, dong.hyun@navercorp.com, qwertgfdcvb@kaist.ac.kr,
wonseok.hwang@lbox.kr, daehyun.nam@upstage.ai, sungrae.park@upstage.ai

Abstract

Key information extraction (KIE) from document images requires understanding the contextual and spatial semantics of texts in two-dimensional (2D) space. Many recent studies try to solve the task by developing pre-trained language models focusing on combining visual features from document images with texts and their layout. On the other hand, this paper tackles the problem by going back to the basic: effective combination of text and layout. Specifically, we propose a pre-trained language model, named *BROS (BERT Relying On Spatiality)*, that encodes relative positions of texts in 2D space and learns from unlabeled documents with area-masking strategy. With this optimized training scheme for understanding texts in 2D space, BROS shows comparable or better performance compared to previous methods on four KIE benchmarks (FUNSD, SROIE*, CORD, and SciTSR) without relying on visual features. This paper also reveals two real-world challenges in KIE tasks–(1) minimizing the error from incorrect text ordering and (2) efficient learning from fewer downstream examples–and demonstrates the superiority of BROS over previous methods. *Code will be available at https://github.com/clovaai.*

Introduction

Automatic key information extraction (KIE) from industrial documents is an essential task in robotic process automation (RPA). Extracting an ordered item list from receipts (Park et al. 2019), prices and taxes from invoices (Liu et al. 2019), and paired key-values from form-like documents (Jaume, Ekenel, and Thiran 2019) are representative examples. Since the task requires understanding texts in various layouts, the combination of multiple technical components from both computer vision and natural language processing is required.

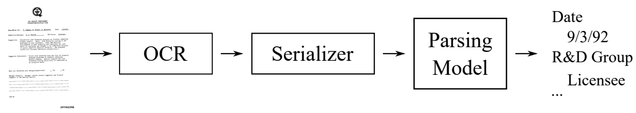


Figure 1: Schematic illustrations of document KIE pipeline.

Model	Img	# Params	(O)	(P)	(F)
$\text{LayoutLM}_{\text{BASE}}$		113M	78.66	33.89	62.50
$\text{LayoutLMv2}_{\text{BASE}}$	○	200M	82.76	40.77	69.92
$\text{BROS}_{\text{BASE}}$		**110M**	**83.05**	**76.94**	**72.60**
$\text{LayoutLM}_{\text{LARGE}}$		343M	78.95	33.11	61.00
$\text{LayoutLMv2}_{\text{LARGE}}$	○	426M	84.20	62.53	72.12
$\text{BROS}_{\text{LARGE}}$		**340M**	**84.52**	**79.42**	**74.42**

Table 1: Performance comparison of pre-trained language models on (O)riginal, (P)ermuted, and (F)ew training samples FUNSD KIE tasks. In (F), 10 samples are used.

Figure 1 describes a schematic illustration of pipeline for the document KIE tasks (Hwang et al. 2019; Denk and Reisswig 2019). First, given a document image, optical character recognition (OCR) detects the texts in the image and recognizes the content to generate a set of text blocks. Next, a serializer identifies a reading order of text blocks distributed in 2D image space and converts them into text sequence in 1D text space to apply NLP technology which is developed for 1D text sequence. The most basic form of serializer is to arrange text blocks in a top-to-bottom and left-to-right way (Clausner, Pletschacher, and Antonacopoulos 2013). Finally, from the serialized text blocks, key information is extracted via the parsing model.

In the first step, an off-the-shelf OCR tool is often employed as industrial documents consist of relatively clean characters compared to general scene text images. On the other hand, there are no such off-the-shelf tools for the serializer even though it is often non-trivial to determine the proper reading order of text blocks (Li et al. 2020; Wang et al. 2021). Representative examples are documents including multi-columns or multiple tables. This absence of the general-purpose serializer implies the careful design of parsing module is necessary to robustly handle documents with complex layouts where the reading orders can be often incomplete.

In the early studies of KIE, accurate document parsing greatly depends on the order of text blocks. Once the serializer identifies an order, the set of the text blocks are converted into a text sequence and processed via a language model such as BERT (Devlin et al. 2019) to identify key information (Hwang et al. 2019; Denk and Reisswig 2019). The linguistic understanding of the pre-trained language model leads to superior performance than rule-based extractions. However, the conversion of texts in 2D space into a text sequence in 1D space leads to the loss of layout information that is critical in KIE tasks.

To avoid the loss of layout information, a new type of language model, LayoutLM (Xu et al. 2020) expands a 1D positional encoding of BERT to 2D and is trained over a large corpus of industrial documents to understand spatial dependencies between text blocks. Its fine-tuning has shown breakthrough performances on multiple KIE tasks and becomes a strong baseline. After the rise of LayoutLM, several studies try to develop pre-trained language models by combining additional visual features (Xu et al. 2021; Powalski et al. 2021; Li et al. 2021b; Appalaraju et al. 2021; Li et al. 2021c) (e.g. image patches identified by an object detection) and show further performance improvements. However, the extensions using visual features require additional computational costs and they still demand more effective combinations of texts and their spatial information.

In this paper, we introduce a new pre-trained language model, named BROS, by re-focusing on the combinations of texts and their spatial information without relying on visual features. Specifically, we propose an effective spatial encoding method by utilizing relative positions between text blocks, while most of previous works employ absolute 2D positions. Additionally, we introduce a novel self-supervision method, named area-masked language model, that hides texts in an area of a document and supervises the masked texts. With these two approaches for encoding of spatial information, BROS shows superior or comparable performances compared to previous methods using additional visual features.

Aside from improving KIE performances, BROS also addresses two important real-world challenges in KIE tasks: minimizing dependency on the order of text blocks and learning from a few training examples of downstream tasks. The first challenge indicates the robustness on the serialization followed by the OCR process in Figure 1. In real scenario, document images are usually irregular (i.e. rotated or distorted documents) and the serializer might fail to identify a proper order of text blocks. In addition, when serialization fails, the performance of sequence tagging approaches (e.g. BIO tagging), which most previous works employ, drops dramatically. To circumvent the difficulty, we apply SPADE (Hwang et al. 2021) decoder that extracts key text blocks without any order information to the pre-trained models and evaluates them on the new benchmarks where the order of text blocks are permuted. As a result, BROS shows better robustness on the serializers compared to LayoutLM (Xu et al. 2020) and LayoutLMv2 (Xu et al. 2021).

The second challenge is related to the required number of labeled examples to understand the target key contents. Since a single KIE example consists of hundreds of text blocks that should be categorized, the annotation is expensive. Most public benchmarks consist of less than 1,000 samples, even though the target documents contain hundreds of layouts and diverse contexts. In this paper, we analyze KIE performances over the number of training examples and compare the pre-trained models. As a result, BROS performs better on FUNSD KIE tasks, and also BROS only with 20$\sim$30% of FUNSD examples achieves better performance than LayoutLM with 100% of them. Summarized results for these experiments are shown in Table 1.

Our contributions can be summarized as follows:

- We propose an effective spatial layout encoding method by accounting for relative positions of text blocks.
- We also propose a novel area-masking self-supervision strategy that reflects 2D natures of text blocks.
- The proposed model achieves comparable performance to the state-of-the-art without relying on visual features.
- We compare existing pre-trained models on permuted KIE datasets that lost the orders of text blocks.
- We compare the fine-tuning efficiency of various pre-trained models under a data-scarce environment.

Related Work

Pre-trained Language Models for 2D Text Blocks

Unlike the pre-trained models for conventional NLP tasks, such as BERT (Devlin et al. 2019), LayoutLM (Xu et al. 2020) is first proposed to jointly model interaction between text and layout information for the document KIE task. It encodes the absolute position of text blocks with axis-wise embedding tables and learns a token-level masked language model that hides tokens randomly and estimates the origins. After the publication of LayoutLM, several pre-trained models have been tried to additionally integrate visual features, such as visual feature maps from raw images (Xu et al. 2021; Appalaraju et al. 2021), image patches identified by an object detection module (Li et al. 2021b), and visual representations of text blocks (Powalski et al. 2021; Li et al. 2021c). Although the extensions imposing multi-modalities of visual and textual features provide additional performance gains in KIE tasks, they spend additional computations to process raw document images. Additionally, an effective combination of text and layout is still required as the major component of the multi-modalities.

Aside from incorporating visual features, StructuralLM (Li et al. 2021a) utilizes cell information, a group of ordered text blocks, and shows promising performance improvements. However, the local orders of text blocks might not be available depending on the KIE tasks and the OCR engines. Therefore, this paper focuses on the original granularity of text blocks identified by OCR engines and improves the combination of text and layout by an effective spatial encoding method and an area-based pre-training strategy.

Parsers for Document Key Information Extraction

BIO tagger, which is a representative parser for entity extraction from the text sequences, extracts key information by identifying spans with the beginning (B) and inside (I) points. Though BIO tagger has been used as a conventional method, it has two limitations for applying to document KIE. One is that the correct order of text blocks is required for extracting key information when post-processing each classified token class (i.e. B- and I- classes). For example, if the text blocks are not ordered properly, such as “recognition,

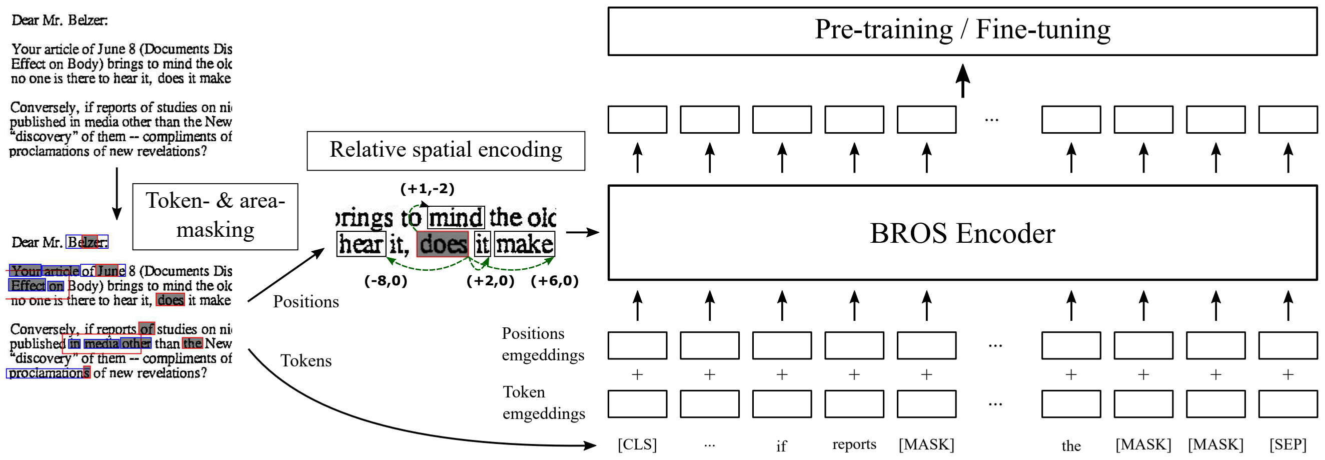

Dear Mr. Belzer:
Your article of June 8 (Documents Dis
Effect on Body) brings to mind the olc
no one is there to hear it, does it make
Conversely, if reports of studies on ni
published in media other than the New
"discovery" of them -- compliments of
proclamations of new revelations?
Token- & area-masking
Relative spatial encoding
(+1,-2)
(-8,0)
(+2,0)
(+6,0)
Positions
Tokens
Pre-training / Fine-tuning
BROS Encoder
Positions emgeddings
Token emgeddings
[CLS]
...
if
reports
[MASK]
the
[MASK]
[MASK]
[SEP]

Figure 2: An overview of BROS. The tokens in the document image are masked through token- and area-masking strategy. The position difference between text blocks is encoded directly to the attention mechanism in Transformer. The output token representations are used in both pre-training and fine-tuning.

optical, character", the correct answer can not be made. The other is that it cannot solve the tasks that require the relationship between tokens as it performs token-level classification.

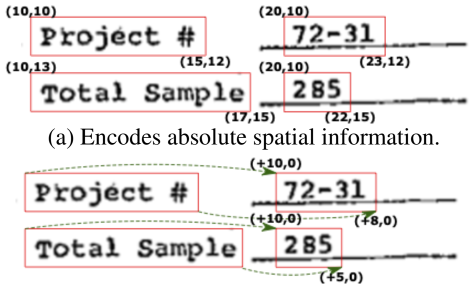
To overcome the above two limitations, we adopt a graph-based parser, SPADE (Hwang et al. 2021) decoder, that creates a directed relation graph of tokens to represent key entities and their relationships for KIE tasks. For example, SPADE can determine "optical" as a starting word and "recognition" as the next word. By directly identifying relations between tokens, SPADE enables a description of all key information of KIE tasks regardless of the order of text blocks. In this paper, we apply the SPADE decoder for entity linking tasks of KIE benchmarks and also for all tasks lost perfect order information of text blocks. Specifically, we slightly modify the SPADE decoder for better application with the pre-trained models.

BERT Relying on Spatiality (BROS)

The main structure of BROS follows LayoutLM (Xu et al. 2020), but there are two critical advances: (1) a use of spatial encoding metric that describes spatial relations between text blocks and (2) a use of 2D pre-training objective designed for text blocks on 2D space. Figure 2 shows a visual description of BROS for document KIE tasks.

Encoding Spatial Information into BERT

The way to encode spatial information of text blocks decides how text blocks be aware of their spatial relations. LayoutLM (Xu et al. 2020) simply encodes absolute x- and y-axis positions to each text blocks but the specific-point encoding is not robust on the minor position changes of text blocks. Instead, BROS employs relative positions between text blocks to explicitly encode spatial relations. As shown in Figure 3, relative positions provides co-modality of spatial relations between text blocks regardless of their absolute position. This property can make the model better recognize entities which have similar key-value structures.



(a) Encodes absolute spatial information.

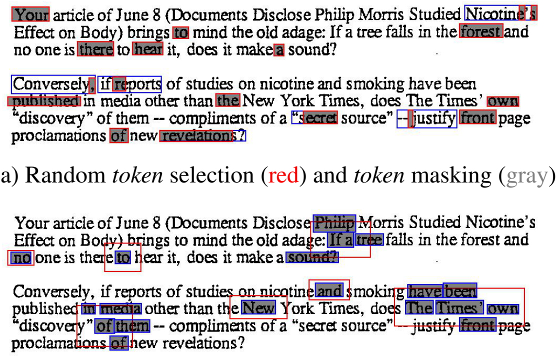
(b) Encodes relative spatial information.

Figure 3: Comparison between absolute and relative positions. "Project #" and "Total Sample" have their paired values, "72-31" and "285", respectively. In (a), the paired text blocks have different modalities based their absolute positions. On the other hand, in (b), they can hold co-modality to represent positions of their semantically coupled text blocks.

For formal description, we use $\boldsymbol{p} = (x, y)$ to denote a point on 2D space and a bounding box of a text block consists of four vertices, such as $\boldsymbol{p}^{\text{tl}}$, $\boldsymbol{p}^{\text{tr}}$, $\boldsymbol{p}^{\text{br}}$, and $\boldsymbol{p}^{\text{bl}}$, that indicate top-left, top-right, bottom-right, and bottom-left points, respectively. BROS first normalizes all the 2D points of the text blocks using the size of the image. Then, BROS calculates relative positions of the vertices from the same vertices of the other bounding boxes of text blocks and applies sinusoidal functions as $\bar{\boldsymbol{p}}_{i,j} = [\mathbf{f}^{\text{sinu}}(x_i - x_j); \mathbf{f}^{\text{sinu}}(y_i - y_j)]$. Here, $\mathbf{f}^{\text{sinu}} : \mathbb{R} \rightarrow \mathbb{R}^{D^{\text{s}}}$ indicates a sinusoidal function, which is used in Vaswani et al. (2017), D^{s} is the dimensions of sinusoid embedding, and the semicolon (;) indicates concatenation. Through the calculations, the relative positions of j^{th} bounding box based on the i^{th} bounding box are represented with the four vectors, such as $\bar{\boldsymbol{p}}^{\text{tl}}_{i,j}$, $\bar{\boldsymbol{p}}^{\text{tr}}_{i,j}$, $\bar{\boldsymbol{p}}^{\text{br}}_{i,j}$, and $\bar{\boldsymbol{p}}^{\text{bl}}_{i,j}$. Finally, BROS combines the four relative positions by applying a linear transformation,

$$\overline{\boldsymbol{bb}}_{i,j} = \boldsymbol{W}^{\text{tl}}\bar{\boldsymbol{p}}^{\text{tl}}_{i,j} + \boldsymbol{W}^{\text{tr}}\bar{\boldsymbol{p}}^{\text{tr}}_{i,j} + \boldsymbol{W}^{\text{br}}\bar{\boldsymbol{p}}^{\text{br}}_{i,j} + \boldsymbol{W}^{\text{bl}}\bar{\boldsymbol{p}}^{\text{bl}}_{i,j}. \quad (1)$$

where $\boldsymbol{W}^{\text{tl}}, \boldsymbol{W}^{\text{tr}}, \boldsymbol{W}^{\text{br}}, \boldsymbol{W}^{\text{bl}} \in \mathbb{R}^{(H/A) \times 2D^{\text{s}}}$ are linear transition matrices, H is a hidden size of BERT, and A is the



Your article of June 8 (Documents Disclose Philip Morris Studied Nicotine's Effect on Body) brings to mind the old adage: If a tree falls in the forest and no one is there to hear it, does it make a sound?

Conversely, if reports of studies on nicotine and smoking have been published in media other than the New York Times, does The Times' own "discovery" of them -- compliments of a "secret source" -- justify front page proclamations of new revelations?

(a) Random *token* selection (red) and *token* masking (gray)

Your article of June 8 (Documents Disclose Philip Morris Studied Nicotine's Effect on Body) brings to mind the old adage: If a tree falls in the forest and no one is there to hear it, does it make a sound?

Conversely, if reports of studies on nicotine and smoking have been published in media other than the New York Times, does The Times' own "discovery" of them -- compliments of a "secret source" -- justify front page proclamations of new revelations?

(b) Random *area* selection (red) and *block* masking (gray)

Figure 4: Illustrations of two masking strategies. The blue boxes represent text blocks including masked tokens. In both figures, 15% of tokens are masked.

number of self-attention heads.

In the process of identifying the relative positional vector, $\overline{\boldsymbol{bb}}_{i,j}$, we carefully apply two components: the sinusoidal function, $\mathbf{f}^{\text{sinu}}$, and the shared embeddings to multiple heads of the attention module. First, the sinusoidal function can encode continuous distances more naturally than using a grid embedding that split a real-valued space into finite number of grids. Second, the multi-head attention modules in Transformer share the same relative positional embeddings to impose the common spatial relationships between text blocks to multiple semantic features identified by the multiple heads.

BROS directly encodes the spatial relations to the contextualization of text blocks. In detail, it calculates an attention logit combining both semantic and spatial features as follows;

$$a^h_{i,j} = (\boldsymbol{W}^{\mathrm{q}}_h \boldsymbol{t}_i)^\top (\boldsymbol{W}^{\mathrm{k}}_h \boldsymbol{t}_j) + (\boldsymbol{W}^{\mathrm{q}}_h \boldsymbol{t}_i)^\top \overline{\boldsymbol{bb}}_{i,j}, \qquad (2)$$

where $\boldsymbol{t}_i$ and $\boldsymbol{t}_j$ are context representations for i^{th} and j^{th} tokens and both $\boldsymbol{W}^{\mathrm{q}}_h$ and $\boldsymbol{W}^{\mathrm{k}}_h$ are linear transition matrices for h^{th} head. The former is the same as the original attention mechanism in Transformer (Vaswani et al. 2017). The latter, motivated by Dai et al. (2019), considers the relative spatial information of the target text block when the source context and location are given. As we mentioned above, we have shared relative spatial embedding across all of the different attention heads for imposing the common spatial relationships.

Compared to the spatial-aware attention in Xu et al. (2021), which utilizes axis-specific positional difference of text blocks as an attention bias, it has two major differences. First, our method couples the relative embeddings with the semantic information of tokens for better conjugation between texts and their spatial relations. Second, when calculating the relative spatial information between two text blocks, we consider all four vertices of the block. By doing this, our encoding can incorporate not only relative distance but also relative shape and size which play important roles in distinguishing key and value in a document. We compare our relative encoding method and that of LayoutLMv2's in the ablation study.

Area-masked Language Model

Pre-training diverse layouts from unlabeled documents is a key factor for document KIE tasks. BROS utilizes two pre-training objectives: one is a token-masked LM (TMLM) used in BERT and the other is a novel area-masked LM (AMLM) introduced in this paper. The area-masked LM, inspired by SpanBERT (Joshi et al. 2020), captures consecutive text blocks based on a 2D area in a document.

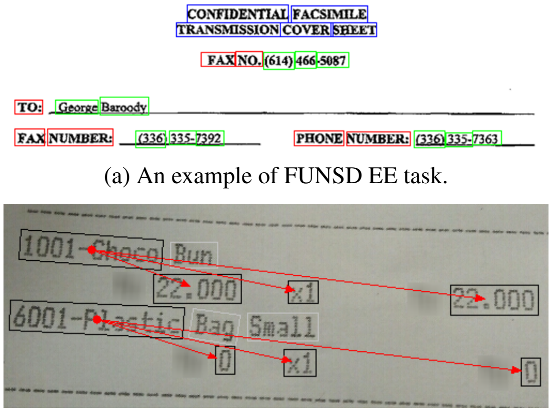
TMLM randomly masks tokens while keeping their spatial information, and then the model predicts the masked tokens with the clues of spatial information and the other un-masked tokens. The process is identical to MLM of BERT and Masked Visual-Language Model (MVLM) of LayoutLM. Figure 4 (a) shows how TMLM masks tokens in a document. Since tokens in a text block can be masked partially, their estimation can be conducted by referring to other tokens in the same block or text blocks near the masked token.

AMLM masks all text blocks allocated in a randomly chosen area. It can be interpreted as a span masking for text blocks in 2D space. Specifically, AMLM consists of the following four steps: (1) randomly selects a text block, (2) identifies an area by expanding the region of the text block, (3) determines text blocks allocated in the area, and (4) masks all tokens of the text blocks and predicts them. At the second step, the degree of expansion is identified by sampling a value from an exponential distribution with a hyper-parameter, λ. The rationale behind using exponential distribution is to convert the geometric distribution used in SpanBERT for a discrete domain into a distribution for a continuous domain. Thus, we set $\lambda = -\ln(1-p)$ where $p = 0.2$ used in SpanBERT. Also, we truncated exponential distribution with 1 to prevent an infinity value covering all spaces of the document. It should be noted that the masking area is expanded from a randomly selected text block since the area should be related to the text sizes and locations to represent text spans in 2D space. Figure 4 compares token- and area-masking on text blocks. Because AMLM hides spatially close tokens together, their estimation requires more clues from text blocks far from the estimation targets.

Finally, BROS combines two masked LMs, TMLM and AMLM, to stimulate the model to learn both individual and consolidated token representations. It first masks 15% of tokens for AMLM and then masks 15% of tokens on the left text blocks for TMLM. Similar to BERT (Devlin et al. 2019), the masked tokens are replaced by `[MASK]` token for 80%, random token for 10%, and original token for the rest 10%.

Key Information Extraction Tasks

We solve two categories of KIE tasks, entity extraction (EE) and entity linking (EL). The EE task identifies sequences of text blocks that represent desired target texts. Figure 5 (a) is an example of the EE task: identifying header, question, and answer entities in the form-like document. The EL task connects key entities through their hierarchical or semantic relations. Figure 5 (b) is an example



(a) An example of FUNSD EE task.

(b) An example of CORD EL task.

Figure 5: Examples of EE and EL tasks. In (a), the colored blocks represent key entities. In (b), the red arrows show the hierarchical relationships between the entities.

Dataset	Types	Tasks	# Images
FUNSD	Forms	EE, EL	Train 149, Test 50
SROIE*†	Receipts	EE	Train 526, Test 100
CORD	Receipts	EE, EL	Train 800, Val 100, Test 100
SciTSR	Tables	EL	Train 12,000, Test 3,000

† modified version of SROIE. See details in Appendix.

Table 2: Tasks and the number of images for each dataset.

of the EL task: grouping menu entities, such as its name, unit price, amount, and price. Table 2 lists four KIE benchmark datasets: FUNSD (Jaume, Ekenel, and Thiran 2019), SROIE* (Huang et al. 2019), CORD (Park et al. 2019), and SciTSR (Chi et al. 2019).

Although these four datasets provide testbeds for the EE and EL tasks, they represent the subset of real problems as the order information of text blocks is given. FUNSD provides the 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 reflect the real scenario that does not contain perfect order information of text blocks, we remove the order information of KIE benchmarks by randomly permuting the order of text blocks. We denote the permuted datasets as p-FUNSD, p-SROIE*, p-CORD, and p-SciTSR.

Experiments

Experiment Settings

For pre-training, IIT-CDIP Test Collection 1.0[1] (Lewis et al. 2006), which consists of approximately 11M document images, is used but 400K of RVL-CDIP dataset[2] (Harley, Ufkes, and Derpanis 2015) are excluded following LayoutLM. To obtain text blocks from document images, CLOVA OCR API[3] was applied. We observed no difference in performance depending on the OCR engine; LayoutLM trained in our experimental setting shows comparable performances to the published LayoutLM.

The main Transformer structure of BROS is the same as BERT. We set the hidden size, the number of self-attention heads, the feed-forward/filter size, and the number of Transformer layers of BROS$_{\text{BASE}}$ to 768, 12, 3072, and 12, respectively and those of BROS$_{\text{LARGE}}$ to 1024, 24, 4096, and 24, respectively. The dimensions of sinusoid embedding D^s is set to 24 for BROS$_{\text{BASE}}$ and 32 for BROS$_{\text{LARGE}}$.

BROS is trained by using AdamW optimizer (Loshchilov and Hutter 2019) with a learning rate of 5e-5 with linear decay. The batch size is set to 64. During pre-training, the first 10% of the total epochs are used for a warm-up learning rate. We initialized weights of BROS with those of BERT and trained it for 5 epochs on the IIT-CDIP dataset using 8 NVIDIA Tesla V100 32GB GPUs.

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.

Experiment Results

To evaluate the performance of the model, we first conduct experiments using the given order of text blocks in the dataset. Then, we verify the robustness of the model against two important challenges in the KIE tasks, which are the dependency about the order of text blocks and learning from a few training examples.

Over our experiments, we report the scores of LayoutLM and LayoutLMv2 using the models published by the authors[4] and denote them as LayoutLM* and LayoutLMv2*. We report the mean (and optionally the standard deviation) of the results using the 5 different random seeds.

With the Order Information of Text Blocks Table 3 summarizes the results for the FUNSD EE task reported by previous approaches. When comparing models only using text and layout, BROS shows remarkable performance improvements by 2.51 (80.54 → 83.05) for the BASE models and 5.57 (78.95 → 84.52) for the LARGE models from the previous best. Interestingly, BROS provides better or similar performances compared to the multi-modal models incorporating additional visual (*Image**) or hierarchical (*Cell**) information. In other words, although BROS does not require extra computations and parameters to process additional features, BROS can achieves better or comparable performances.

Table 4 shows the F1 scores on three EE and EL tasks with the order of text blocks given in the dataset. F, S, C, and Sci refer to FUNSD, SROIE*, CORD, and SciTSR, respectively. For EE tasks, all models utilize BIO tagger that captures spans of text blocks to represent key entities in documents. For EL tasks, SPADE decoder is used to identify relationships between entities not placed sequentially in a series of text blocks. In all cases, BERT performs the worst because those tasks require understanding texts in 2D space, but

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

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

[3] https://clova.ai/ocr

[4] https://github.com/microsoft/unilm

Model	Modality	FUNSD EE			# Params
		Precision	Recall	F1	
BERT$_{\text{BASE}}$ (Xu et al. 2020)	Text	54.69	67.10	60.26	110M
LayoutLM$_{\text{BASE}}$ (Xu et al. 2020)	Text + Layout	75.97	81.55	78.66	113M
DocFormer$_{\text{BASE}}$ (Appalaraju et al. 2021)	Text + Layout	77.63	83.69	80.54	149M
BROS$_{\text{BASE}}$ (Ours)	Text + Layout	**81.16**±0.33	**85.02**±0.32	**83.05**±0.26	110M
LayoutLM$_{\text{BASE}}$ (Xu et al. 2020)	Text + Layout + *Image**	76.77	81.95	79.27	160M
LayoutLMv2$_{\text{BASE}}$ (Xu et al. 2021)	Text + Layout + *Image**	80.29	85.39	82.76	200M
DocFormer$_{\text{BASE}}$ (Appalaraju et al. 2021)	Text + Layout + *Image**	80.76	86.09	83.34	183M
SelfDoc (Li et al. 2021b)	Text + Layout + *Image**	-	-	83.36	137M
StrucTexT (Li et al. 2021c)	Text + Layout + *Image**	85.68	80.97	83.09	107M†
BERT$_{\text{LARGE}}$ (Xu et al. 2020)	Text	61.13	70.85	65.63	340M
LayoutLM$_{\text{LARGE}}$ (Xu et al. 2021)	Text + Layout	75.96	82.19	78.95	343M
BROS$_{\text{LARGE}}$ (Ours)	Text + Layout	**82.81**±0.35	**86.31**±0.28	**84.52**±0.30	340M
LayoutLMv2$_{\text{LARGE}}$ (Xu et al. 2021)	Text + Layout + *Image**	83.24	85.19	84.20	426M
StructuralLM$_{\text{LARGE}}$ (Li et al. 2021a)	Text + Layout + *Cell**	83.52	86.81	85.14	355M

† The number of parameters except for ResNet-FPN processing document images.

Table 3: Performance comparison on the FUNSD EE task. **Bold** indicates the best performance among models using only text and layout, and underline represents the best one. *Image** and *Cell** denote additional visual and hierarchical information, respectively. Our methods are repeatedly evaluated five times and the values of other methods are the reported scores.

Model	Entity Extraction			Entity Linking		
	F	S	C	F	C	Sci
BERT$_{\text{BASE}}$	60.92	93.67	93.13	27.65	92.83	86.76
LayoutLM$^*_{\text{BASE}}$	78.54	95.11	96.26	45.86	95.21	99.05
LayoutLMv2$^*_{\text{BASE}}$	81.89	96.09	96.05	42.91	95.59	98.19
BROS$_{\text{BASE}}$	**83.05**	**96.28**	**96.50**	**71.46**	**95.73**	**99.45**
BERT$_{\text{LARGE}}$	64.17	94.25	94.74	29.11	94.31	89.23
LayoutLM$^*_{\text{LARGE}}$	79.27	95.36	96.12	42.83	95.41	99.33
LayoutLMv2$^*_{\text{LARGE}}$	83.59	96.39	97.24	70.57	97.29	**99.76**
BROS$_{\text{LARGE}}$	**84.52**	**96.62**	**97.28**	**77.01**	**97.40**	99.58

Table 4: Performance comparisons on three EE and EL tasks *with* the order information of text blocks.

Model	Entity Extraction			Entity Linking		
	p-F	p-S	p-C	p-F	p-C	p-Sci
BERT$_{\text{BASE}}$	18.85	39.73	59.71	9.59	27.88	1.75
LayoutLM$^*_{\text{BASE}}$	33.89	66.05	80.86	22.98	61.51	97.32
LayoutLMv2$^*_{\text{BASE}}$	40.77	73.56	80.37	23.25	50.55	95.86
BROS$_{\text{BASE}}$	**76.94**	**82.85**	**95.86**	**69.61**	**87.72**	**99.19**
BERT$_{\text{LARGE}}$	18.10	43.19	57.17	10.81	27.12	1.93
LayoutLM$^*_{\text{LARGE}}$	33.11	56.84	82.88	20.72	61.98	97.64
LayoutLMv2$^*_{\text{LARGE}}$	62.53	84.92	94.43	50.14	85.80	**99.45**
BROS$_{\text{LARGE}}$	**79.42**	**85.14**	**96.81**	**75.61**	**90.49**	99.33

Table 5: Performance comparisons on three EE and EL tasks *without* the order information of text blocks.

Model	p-	xy-	yx-	original
LayoutLM$^*_{\text{BASE}}$	33.89	34.02	55.47	78.47
LayoutLMv2$^*_{\text{BASE}}$	40.77	52.08	62.37	78.16
BROS$_{\text{BASE}}$	**76.94**	**77.16**	**77.42**	**81.61**
LayoutLM$^*_{\text{LARGE}}$	33.11	33.54	41.45	48.30
LayoutLMv2$^*_{\text{LARGE}}$	62.53	69.14	75.45	83.00
BROS$_{\text{LARGE}}$	**79.42**	**79.91**	**80.02**	**83.23**

Table 6: Comparison of FUNSD EE performances according to sorting methods.

BERT only encodes 1D sequential information. LayoutLM* and LayoutLMv2* show better performance than BERT since they encode layout features as well as text features. And by combining visual features, LayoutLMv2* performs better than LayoutLM* in most tasks. BROS shows the best performance in all tasks except SciTSR. It should be noted that the BROS$_{\text{BASE}}$ show better performance than that of LayoutLM$^*_{\text{LARGE}}$, even though it uses three times lower number of parameters (110M vs 343M). These results indicate that BROS effectively encodes the text and layout features.

Without the Order Information of Text Blocks As we mentioned in previous section, we introduce the permuted KIE benchmarks lost the orders of text blocks by shuffling the provided orders. To solve EE and EL tasks without the order information, we employ the SPADE decoder for all tasks. Table 5 shows the comparison results. p-F, p-S, p-C, and p-Sci refer to p-FUNSD, p-SROIE*, p-CORD, and p-SciTSR, respectively. BERT, which does not employ any spatial information of text blocks, shows the worst results on the orderless conditions. By being aware of the spatiality, layout-aware language models show better performances than BERT, and BROS achieves the best except p-SciTSR. More interestingly, BROS shows minor performance drops compared to Table 4, while LayoutLM* and LayoutLMv2* suffer from huge performance degradations by losing the order information of text blocks.

To systematically investigate how the order information

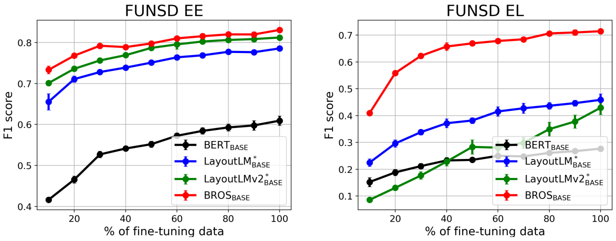


Figure 6: Performance comparisons according to the amount of fine-tuning data. Each point represents the result of fine-tuning using from 10% to 100% of training data.

Dataset	# Data	BERT	LayoutLM*	LayoutLMv2*	BROS
FUNSD EE	5	31.51	48.23	64.26	**68.35**
	10	40.46	62.50	69.92	**72.60**
FUNSD EL	5	14.65	21.38	7.32	**31.11**
	10	14.88	21.48	13.99	**39.17**

Table 7: Results of training with 5 and 10 examples.

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 the 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 6 shows performance on p-FUNSD, xy-FUNSD, yx-FUNSD, and the original FUNSD with the SPADE decoder. In our experiment, LayoutLM$^*_{\text{LARGE}}$ achieves unstable performance (48.30±12.51), when combined with SPADE decoder. Interestingly, the performances of LayoutLM* and LayoutLMv2* are degraded in the order of FUNSD, yx-FUNSD, xy-FUNSD, and p-FUNSD as like the order of the reasonable serialization for text on 2D space. On the other hand, the performance of BROS is relatively consistent. These results show the robustness of BROS on multiple types of serializers.

Learning from Few Training Examples One of the advantages of pre-trained models is that it shows effective transfer learning performance even with a few training examples (Devlin et al. 2019). Since collecting fine-tuning data requires a lot of resource, achieving high performance with a small number of training examples is important.

Figure 6 shows the results of the FUNSD KIE tasks by varying the amount of training examples from 10% to 100% during fine-tuning. In all models, performances tend to increase as the ratio of training data increased. In both tasks, BROS shows the best performances regardless of the number of training samples.

To further test extreme cases, we conduct experiments using only 5 and 10 training examples. Table 7 shows the results of the FUNSD KIE tasks. We fine-tune models for 100 epochs with a batch size of 4. In all cases, BROS shows the best performances. The results prove the generalization ability of BROS even when there are very few training examples.

	Entity Extraction			Entity Linking		
Model	F	S	C	F	C	Sci
LayoutLM$^{\dagger}_{\textbf{BASE}}$	76.89	94.99	94.37	44.00	93.60	99.06
→ *pos enc. only*	78.84	95.45	96.36	59.92	94.83	99.22
→ *objectives only*	78.44	94.81	95.95	47.22	94.11	99.20
→ *both* (= **BROS$_{\text{BASE}}$**)	**80.58**	**95.72**	**96.64**	**65.24**	**96.03**	**99.28**

Table 8: Performance improvements on EE and EL tasks through adding components of BROS. At the last line, all components are changed from LayoutLM and the model becomes BROS.

	Entity Extraction			Entity Linking		
SE	F	S	C	F	C	Sci
Absolute	78.44	94.81	95.95	47.22	94.11	99.20
LayoutLMv2's	78.93	94.71	95.82	53.57	95.27	**99.28**
Ours	**80.58**	**95.72**	**96.64**	**65.24**	**96.03**	**99.28**

Table 9: Spatial encoding methods from BROS' setting.

Ablation Study

We conduct ablation studies to investigate which component contributes the performance improvement. For the ablation studies, we utilize LayoutLM† that is our own implementation of LayoutLM for fair comparisons under the same experimental settings. All models in these studies are pre-trained for 1 epoch.

Table 8 provides performance changes from adding our proposed components. When applying our proposed positional encoding to LayoutLM, the performances consistently increase with huge margins of 3.62pp on average over all tasks. Independently, our extension on pre-training objectives solely provides 1.14pp of performance improvement on average. By utilizing both, BROS$_{\text{BASE}}$ provides the best performances with margins of 5.10pp on average. This ablation study proves that each component of BROS solely contributes to performance improvements as well as their combination provides better results.

Table 9 compares three positional encoding methods: absolute position in LayoutLM, relative position in LayoutLMv2, and ours. Relative position methods perform better than absolute one and the performance gap becomes larger in EL tasks. And among them, our method shows the best results.

Conclusion

We propose a pre-trained language model, BROS, which focuses on modeling text and layout features for effective key information extraction from documents. By encoding texts in 2D space with their relative positions and pre-training the model with the area-masking strategy, BROS shows superior performance without relying on any additional visual features. In addition, under the two real-world settings–imprecise text serialization and small amount of training examples–BROS shows robust performance while other models show significant performance degradation.

Acknowledgments

We thank many colleagues at NAVER CLOVA for their help, in particular Yoonsik Kim, Moonbin Yim, Han-cheol Cho, Bado Lee, Seunghyun Park, and Youngmin Baek for useful discussions.

References

Appalaraju, S.; Jasani, B.; Kota, B. U.; Xie, Y.; and Manmatha, R. 2021. DocFormer: End-to-End Transformer for Document Understanding. *arXiv preprint arXiv:2106.11539*.

Chi, Z.; Huang, H.; Xu, H.-D.; Yu, H.; Yin, W.; and Mao, X.-L. 2019. Complicated table structure recognition. *arXiv preprint arXiv:1908.04729*.

Clausner, C.; Pletschacher, S.; and Antonacopoulos, A. 2013. The significance of reading order in document recognition and its evaluation. In *2013 12th International Conference on Document Analysis and Recognition (ICDAR)*, 688–692. IEEE.

Dai, Z.; Yang, Z.; Yang, Y.; Carbonell, J. G.; Le, Q.; and Salakhutdinov, R. 2019. Transformer-XL: Attentive Language Models beyond a Fixed-Length Context. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*.

Denk, T. I.; and Reisswig, C. 2019. BERTgrid: Contextualized Embedding for 2D Document Representation and Understanding. In *Workshop on Document Intelligence at NeurIPS 2019*.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2019. 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)*, 4171–4186.

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

Huang, Z.; Chen, K.; He, J.; Bai, X.; Karatzas, D.; Lu, S.; and Jawahar, C. 2019. ICDAR2019 competition on scanned receipt ocr and information extraction. In *Proceedings of the 15th International Conference on Document Analysis and Recognition (ICDAR)*, 1516–1520. IEEE.

Hwang, W.; Kim, S.; Seo, M.; Yim, J.; Park, S.; Park, S.; Lee, J.; Lee, B.; and Lee, H. 2019. Post-OCR parsing: building simple and robust parser via BIO tagging. In *Workshop on Document Intelligence at NeurIPS 2019*.

Hwang, W.; Yim, J.; Park, S.; Yang, S.; and Seo, M. 2021. Spatial Dependency Parsing for Semi-Structured Document Information Extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, 330–343.

Jaume, G.; Ekenel, H. K.; and Thiran, J.-P. 2019. FUNSD: A dataset for form understanding in noisy scanned documents. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 2, 1–6. IEEE.

Joshi, M.; Chen, D.; Liu, Y.; Weld, D. S.; Zettlemoyer, L.; and Levy, O. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics (TACL)*, 8: 64–77.

Lewis, D.; Agam, G.; Argamon, S.; Frieder, O.; Grossman, D.; and Heard, J. 2006. 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)*, 665–666.

Li, C.; Bi, B.; Yan, M.; Wang, W.; Huang, S.; Huang, F.; and Si, L. 2021a. StructuralLM: Structural Pre-training for Form Understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, 6309–6318.

Li, L.; Gao, F.; Bu, J.; Wang, Y.; Yu, Z.; and Zheng, Q. 2020. 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)*.

Li, P.; Gu, J.; Kuen, J.; Morariu, V. I.; Zhao, H.; Jain, R.; Manjunatha, V.; and Liu, H. 2021b. SelfDoc: Self-Supervised Document Representation Learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 5652–5660.

Li, Y.; Qian, Y.; Yu, Y.; Qin, X.; Zhang, C.; Liu, Y.; Yao, K.; Han, J.; Liu, J.; and Ding, E. 2021c. StrucTexT: Structured Text Understanding with Multi-Modal Transformers. *arXiv preprint arXiv:2108.02923*.

Liu, X.; Gao, F.; Zhang, Q.; and Zhao, H. 2019. 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)*, 32–39.

Loshchilov, I.; and Hutter, F. 2019. Decoupled Weight Decay Regularization. In *Proceedings of the 7th International Conference on Learning Representations (ICLR)*.

Park, S.; Shin, S.; Lee, B.; Lee, J.; Surh, J.; Seo, M.; and Lee, H. 2019. CORD: A Consolidated Receipt Dataset for Post-OCR Parsing. In *Workshop on Document Intelligence at NeurIPS 2019*.

Powalski, R.; Borchmann, Ł.; Jurkiewicz, D.; Dwojak, T.; Pietruszka, M.; and Pałka, G. 2021. Going Full-TILT Boogie on Document Understanding with Text-Image-Layout Transformer. *arXiv preprint arXiv:2102.09550*.

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems 30 (NeurIPS)*, 5998–6008.

Wang, Z.; Xu, Y.; Cui, L.; Shang, J.; and Wei, F. 2021. LayoutReader: Pre-training of Text and Layout for Reading Order Detection. arXiv:2108.11591.

Xu, Y.; Li, M.; Cui, L.; Huang, S.; Wei, F.; and Zhou, M. 2020. 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)*, 1192–1200.

Xu, Y.; Xu, Y.; Lv, T.; Cui, L.; Wei, F.; Wang, G.; Lu, Y.; Florencio, D.; Zhang, C.; Che, W.; et al. 2021. LayoutLMv2: Multi-modal Pre-training for Visually-Rich Document Understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (ACL-IJCNLP)*, 2579–2591.