SelfDoc: Self-Supervised Document Representation Learning

Peizhao Li[1]*, Jiuxiang Gu[2], Jason Kuen[2], Vlad I. Morariu[2], Handong Zhao[2],
Rajiv Jain[2], Varun Manjunatha[2], Hongfu Liu[1]
[1]Brandeis University, [2]Adobe Research
{peizhaoli,hongfuliu}@brandeis.edu
{jigu,kuen,morariu,hazhao,rajijain,vmanjuna}@adobe.com

Abstract

We propose SelfDoc, a task-agnostic pre-training framework for document image understanding. Because documents are multimodal and are intended for sequential reading, our framework exploits the positional, textual, and visual information of every semantically meaningful component in a document, and it models the contextualization between each block of content. Unlike existing document pre-training models, our model is coarse-grained instead of treating individual words as input, therefore avoiding an overly fine-grained with excessive contextualization. Beyond that, we introduce cross-modal learning in the model pre-training phase to fully leverage multimodal information from unlabeled documents. For downstream usage, we propose a novel modality-adaptive attention mechanism for multimodal feature fusion by adaptively emphasizing language and vision signals. Our framework benefits from self-supervised pre-training on documents without requiring annotations by a feature masking training strategy. It achieves superior performance on multiple downstream tasks with significantly fewer document images used in the pre-training stage compared to previous works.

1. Introduction

Documents, such as business forms, scholarly and news articles, invoices, letters, and text-based emails, encode and convey information through language, visual content, and layout structure. Automated document understanding is a crucial research area for business and academic values. It can significantly reduce labor-intensive document workflows through automated entity recognition, document classification, semantic extraction, document completion, *etc*.

Many works have been proposed applying machine learning for document analysis [13, 15, 37, 36, 15, 18]. However, parsing a document remains non-trivial and poses multiple challenges. One challenge is modeling and understanding contextual information when interpreting content. For example, since information in documents is organized for sequential reading, the interpretation of a piece of content relies heavily on its surrounding context. Similarly, a heading can indicate and summarize the meaning of subsequent blocks of text, and a caption could be useful for understanding a related figure. Another challenge is effectively incorporating the cues from multiple data modalities. In contrast to other data formats like images or plain text, documents combine textual and visual information, and both of the two modalities are complemented by the document layout. Additionally, from a practical perspective, many tasks related to document understanding are label-scarce. A framework that can learn from unlabeled documents (*i.e.,* pre-training) and perform model fine-tuning for specific downstream applications is more preferred than the one that requires fully-annotated training data.

In this work, we develop a task-agnostic representation learning framework for document images. Our model fully exploits the textual, visual, and positional information of every semantically meaningful component in a document, *e.g.*, text block, heading, and figure. To model the internal relationships among components in documents, we adopt the contextualized attention mechanism from natural language processing (NLP) [32] and employ it at the component level. We design two branches separately for textual and visual representation learning, and later encourage cross-modal learning with the proposed cross-modality encoder. In order to seek a better modality fusion for downstream usage, we propose a modality-adaptive attention mechanism to fuse the language and vision features adaptively. Moreover, our framework learns a generic representation from a collection of unlabeled documents via self-supervised learning, and afterward, it will be fine-tuned on various document-related downstream applications.

There are two major differences between our SelfDoc and LayoutLM [36], which also introduces a task-agnostic document pre-training framework by applying 2D posi-

*This work was done during the author's internship at Adobe Research.

tional encoding to BERT model [9]. 1) Instead of using *word* as the basic unit for model input, we adopt semantically meaningful components (*e.g.*, *text block*, *heading*, *figure*) as the model input. In a document, a single word can be understood within the local context where it is found, and does not always require analyzing the entire page for every word. For instance, an answer in a questionnaire tends to be a complete sentence and already delivers semantics. Introducing the contextualization between every single word in documents may be redundant and also ignore localized context; 2) We advance the interaction between language and vision modality in the model's pre-training stage, therefore our model can efficiently leverage the multimodal information from unlabeled data. Comparatively, LayoutLM only considers a single modality in the pre-training stage and incorporates the visual clues during the fine-tuning phase.

We evaluate our model on three downstream tasks: document entity recognition, document classification, and document clustering. With the help of our pre-training method, we achieve leading performance on these applications over other pre-training and task-specific models. In short, our work contributes to the advancement of document analysis and intelligence by 1) introducing SelfDoc, a novel task-agnostic self-supervised learning framework for document data. Our model establishes the contextualization over a block of content and involves multimodal information; 2) modeling information from multiple modalities via cross-modal learning in the pre-training stage, and proposing a modality-adaptive attention mechanism to fuse language and vision features for downstream usages; 3) demonstrating superior performance by using fewer samples for pre-training. SelfDoc achieves surpassing performance on multiple downstream tasks comparing to other methods.

2. Related Work

Document Image Understanding. Artificial neural networks have been extensively applied to document analysis and recognition tasks like page segmentation, text location detection, and region labeling. Marinai *et al.* [22] survey connectionist-based approaches on document image processing. Moreover, Michael *et al.* [29] propose a grammatical model for hierarchical document segmentation and labeling, while Hao *et al.* [33] utilize some statistical machine learning approaches to detect physical structures from historical documents. Recently, with deep learning showing great performance in many domains such as computer vision and NLP, some researchers have applied deep learning to document analysis [15, 18, 37, 13, 7]. By utilizing the graphical property of documents, Katti *et al.* [15] employ convolutional neural networks to recognize the bounding box and semantic segmentation in a document. Despite working with visual clues only, Yang *et al.* [37] use both convolutional neural networks and traditional textual embedding techniques to learn scanned documents by self-reconstruction. On handling the inner connections between text segments in invoices, Liu *et al.* [18] apply graph neural networks and manually build edges by similarity to model the inner-relations in documents. Jain and Wigington [13] propose a multimodal ensemble approach combining language and vision models for document classification. These pioneering attempts toward applying machine learning to document data, though exciting and motivating, are heavily task-specific in their model design and require exhaustive annotations for document image representation learning.

Self-Supervised Learning. Pre-training models in NLP have shown great success in producing generic language representation that learns from a large scale of the unlabeled corpus. BERT [9], which stands for bidirectional encoder representations from Transformers [32], and other pre-training models [24, 19, 26, 11] have delivered promising performance on a series of downstream linguistic tasks such as question answering, sentence classification, and named entity recognition. The core idea of BERT is to learn a contextualized representation from corpus intrinsically via two self-supervised strategies. Given its success in NLP, some works extend the Transformer framework and model pre-training to vision-language learning [21, 31]. These works focus on natural images and corresponding textual descriptions, learning the cross-modality alignment between visual and linguistic information, and can be applied to visual question answering [31], referring expression (localize an object with the given referring expression) [21], and image retrieval [11]. Although the cross-modality design in vision-language learning is used as an inspiration for document representation learning, it cannot be directly adapted for document data due to the great differences between document data and natural image data.

Document Pre-training. Most recently, some works have started pre-training models on document images [36, 25]. The first one, LayoutLM [36], inherits the main idea from BERT while receiving the extra positional information for text in documents, and additionally includes image embeddings in the fine-tuning phase. Pramanik *et al.* [25] use Longformer [6] for heavily-word documents and extend the pre-training strategies to multi-page document pre-training. In [7], they introduce a document pre-training method by solving jigsaw puzzles and doing multimodal learning via topic modeling. Although they employ both image and text modalities during the training process, only image information is used when tested. In contrast to these works, we establish our representation learning at the semantic-component level instead of the single word or character level in documents. By learning feature embedding on document components, we avoid the excessive contextualized learning between every word in a document but exploit the relations between each component. Beyond that, we intro-

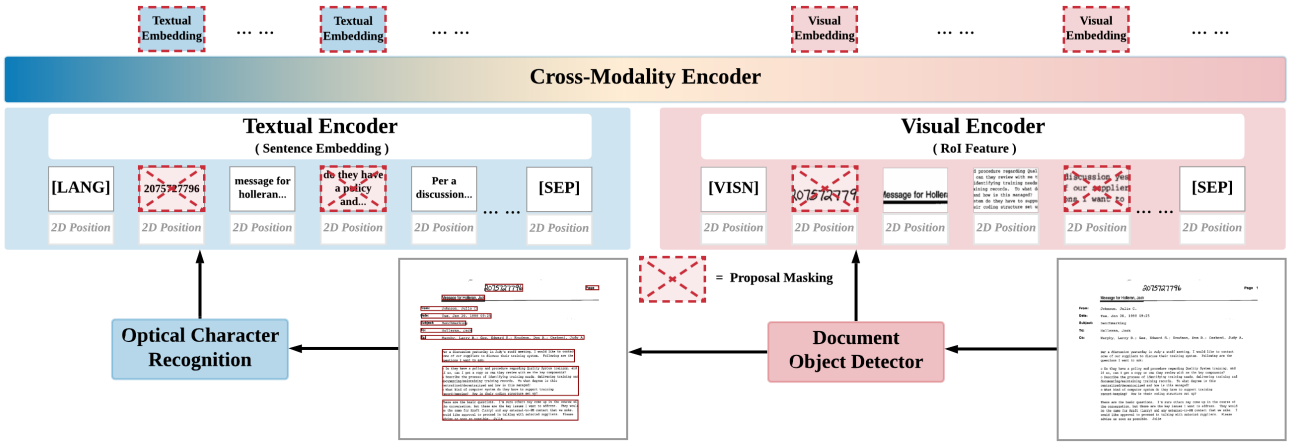

Figure 1. Overview of the proposed document representation learning framework. Extracted language and vision features with corresponding positional encoding are fed into a textual encoder, a visual encoder, and a cross-modal encoder to manipulate the contextual clues and multimodal information within documents. The produced features can be used for document analysis tasks.

duce cross-modality learning in the pre-training phase for contextualized comprehension on document components across language and vision, and leverage multimodal information from document images without annotation.

3. Methodology

Fig. 1 shows an overview of our SelfDoc representation learning framework. It takes document object proposals from a document object detector, and extracts features from both textual and visual modalities with positional encoding to serve as input. For each modality, we employ a single-modality encoder for contextual learning, and later perform learning over the two modalities using the proposed cross-modality encoder. The generated representation can be further utilized for downstream document understanding tasks, such as entity recognition or document classification.

3.1. Pre-processing and Feature Extraction

To begin with, we train a document object detector using Faster R-CNN [28] on public document datasets[38, 23] with bounding box annotations on semantically meaningful components, and localize significant components (*i.e.*, document object proposals) of a document. In our current implementation, we detect the following categories: *text block*, *title*, *list*, *table*, and *figure*. We deem the detected proposals as the basic input unit of our framework. Next, we apply an Optical Character Recognition (OCR) engine [16] to process each cropped proposal from the original document and get the detected text in a default word order.

We then extract the textual and visual features for each proposal. For textual features, we embed plain text contained in a proposal into a feature vector using the pre-trained Sentence-BERT model [27]: a sentence and sequential word learning model that demonstrates superior performance on semantic textual similarity and sentence classification tasks. We extract visual features from Regions-of-Interest (RoI) heads in Faster R-CNN model for every detected proposal. The RoI head uses an adaptive pooling function to output a fixed size vector for proposals of arbitrary sizes. Formally, a document $D = \{p_1, \ldots, p_N\}$ consists of N document object proposals, where each object proposal $p_i = \{x^i_{\text{pos}} \in \mathbb{R}^4, x^i_{\text{visn}} \in \mathbb{R}^{d_{\text{visn}}}, x^i_{\text{lang}} \in \mathbb{R}^{d_{\text{lang}}}\}$ is represented by its 2D coordinate x_{pos}, its RoI feature x_{visn}, and sentence embedding for text x_{lang}, with corresponding feature dimensions d_{lang} and d_{visn} respectively.

Compared to word-level input, the component-level formulation can reduce the input sequence length for a document, especially for text-heavy documents such as scholarly articles. Therefore it decreases the amount of time needed for training and inference since the time complexity for a fully contextualized attention operation (that will be described later) scales quadratically with the input length.

3.2. Input Modeling

Inspired by BERT [9], we mark the beginning of a sentence sequence with a special [LANG] token and an RoI region sequence with [VISN] token (shown in Fig. 1), which are respectively calculated by averaging the sentence features and RoI features. Also, we manually set the positional coordinate of these special proposals to cover the whole document page. The input sequence is then zero-padded to match with its batch-peer for batch training. Then, to incorporate positional information, the input features are mapped to hidden states $\boldsymbol{H}^0_T = \{h^1_T, \ldots, h^N_T\}$ and $\boldsymbol{H}^0_V = \{h^1_V, \ldots, h^N_V\}$ by a linear mapping as follows:

$$h^i_T = \boldsymbol{W}_T x^i_{\text{lang}} + \boldsymbol{W}_P x^i_{\text{pos}},\ h^i_V = \boldsymbol{W}_V x^i_{\text{visn}} + \boldsymbol{W}_P x^i_{\text{pos}}, \quad (1)$$

where matrices $\boldsymbol{W}_T \in \mathbb{R}^{d_h \times d_{\text{lang}}}$, $\boldsymbol{W}_V \in \mathbb{R}^{d_h \times d_{\text{visn}}}$, $\boldsymbol{W}_P \in \mathbb{R}^{d_h \times 4}$ project features into hidden-state in d_h dimension.

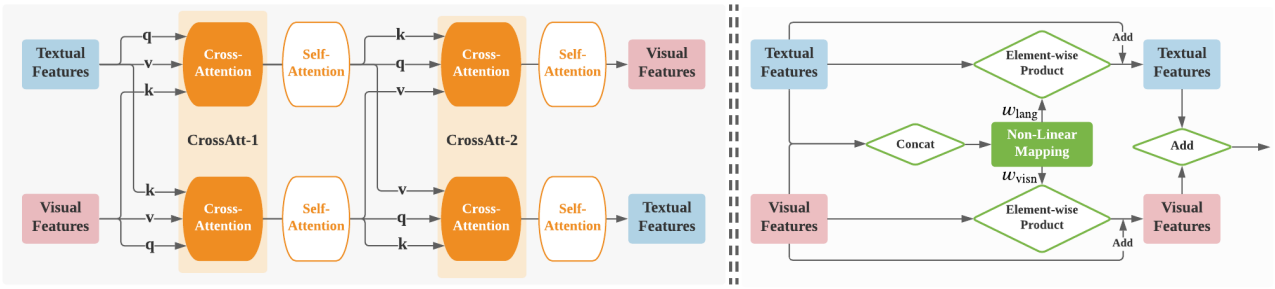

Figure 2. Schematic illustrations of (A): cross-modality encoder with different cross-attention functions inserted sequentially, and (B): modality-adaptive attention used in the fine-tuning phase for adaptively fusing features from language and vision.

3.3. Single-Modality Encoder

Then, the language and vision features are separately passed through the textual and visual encoder. These two encoders follow the same design of a basic module in BERT, but their parameters are not shared. This module contains multi-head attention, feed-forward (FF) layers, residual connection, and layer normalization (LN) [5]. In the multi-layer single-modality encoder, let $\boldsymbol{H}^l = \{h_1, \ldots, h_N\}$ be the encoded features at the l-th layer. $\boldsymbol{H}^0$ is the vector of the input features given in Sec. 3.2. Features output by the next layer $\boldsymbol{H}^{l+1}$ can be obtained via:

$$\boldsymbol{H}^l_{\text{att}} = \text{LN}(f_{\text{SelfAtt}}(\boldsymbol{H}^l) + \boldsymbol{H}^l), \tag{2}$$

$$\boldsymbol{H}^{l+1} = \text{LN}(f_{\text{FF}}(\boldsymbol{H}^l_{\text{att}}) + \boldsymbol{H}^l_{\text{att}}), \tag{3}$$

where $f_{\text{SelfAtt}}(\cdot)$ is the self-attention function defined as

$$f_{\text{SelfAtt}}(\boldsymbol{H}^l) = \text{softmax}\left(\frac{q(\boldsymbol{H}^l)k(\boldsymbol{H}^l)^\top}{\sqrt{d_k}}\right)v(\boldsymbol{H}^l), \tag{4}$$

where $q(\cdot)$, $k(\cdot)$, and $v(\cdot)$ are linear transformation layers applied to features of proposals and they are the query, key, and value, respectively. d_k is the number of attention heads for normalization. For technical details on multi-head attention please refer to [32]. Finally, $\boldsymbol{H}^{l+1}$ can be obtained by $\boldsymbol{H}^l_{\text{att}}$ via a feed-forward sub-layer composed of two fully-connected layers of function $f_{\text{FF}}(\cdot)$. Hierarchically stacked layers form the textual and visual encoders.

The textual and visual encoders produce contextually embedded features for each proposal using the features from surrounding proposals in their respective modality. The vector multiplication between query and key explores similar patterns in sequential proposals and emphasizes the shared part. The outputs of the textual and visual encoder are subsequently fed into the cross-modal encoder described below, which is more focused on cross-modality learning to bridge the multimodal information in language and vision.

3.4. Cross-Modality Encoder

We encourage cross-modality learning by introducing two interactive cross-attention functions. The structure of the cross-modal encoder is similar to the textual or visual encoder, but we substitute the self-attention function $f_{\text{SelfAtt}}(\cdot)$ in Eq. (2) with $f_{\text{CrossAtt-1}}(\cdot)$ or $f_{\text{CrossAtt-2}}(\cdot)$ as elaborated in what follows. We add subscripts T and V to denote the modality in $\boldsymbol{H}^l_{\text{T}}$ and $\boldsymbol{H}^l_{\text{V}}$ which are the intermediate textual and visual representations, respectively.

The first attention function identifies the agreement between language and vision information. At a high level, if the font size or character style in a proposal is confirmed by the semantic meaning of language features, these features should be amplified. Formally, we have

$$f_{\text{CrossAtt-1}}(\boldsymbol{H}^l_{\text{T}}) = \text{softmax}\left(\frac{q(\boldsymbol{H}^l_{\text{T}})k(\boldsymbol{H}^l_{\text{V}})^\top}{\sqrt{d_k}}\right)v(\boldsymbol{H}^l_{\text{T}}), \tag{5}$$

$$f_{\text{CrossAtt-1}}(\boldsymbol{H}^l_{\text{V}}) = \text{softmax}\left(\frac{q(\boldsymbol{H}^l_{\text{V}})k(\boldsymbol{H}^l_{\text{T}})^\top}{\sqrt{d_k}}\right)v(\boldsymbol{H}^l_{\text{V}}). \tag{6}$$

The second attention function serves as the operation to discover inner-relationships from one modality to another. Since documents are naturally composed of two modalities, we have the same number of proposals in language and vision branches, and the input sequences of these two modalities are identically ordered[1] in our input. Based on this, the contextual clues can be propagated between modalities, for instance, the similarity in font style can enhance the understanding of semantic meaning between proposals. We have

$$f_{\text{CrossAtt-2}}(\boldsymbol{H}^l_{\text{T}}) = \text{softmax}\left(\frac{q(\boldsymbol{H}^l_{\text{V}})k(\boldsymbol{H}^l_{\text{V}})^\top}{\sqrt{d_k}}\right)v(\boldsymbol{H}^l_{\text{T}}), \tag{7}$$

$$f_{\text{CrossAtt-2}}(\boldsymbol{H}^l_{\text{V}}) = \text{softmax}\left(\frac{q(\boldsymbol{H}^l_{\text{T}})k(\boldsymbol{H}^l_{\text{T}})^\top}{\sqrt{d_k}}\right)v(\boldsymbol{H}^l_{\text{V}}). \tag{8}$$

[1] Identically ordered means the two modalities have the same input order, *e.g.*, the first index of textual and visual input corresponds to the same proposal.

Note that we do not distinguish the linear transformation query, key, and value for notational simplicity, but they are not shared and are specific for a certain attention function in implementation. We build the cross-modality encoder by alternatively inserting these two types of cross-attention layer and a self-attention layer. A schematic illustration for a cross-modality encoder is presented in Fig. 2(A).

3.5. Pre-training

Our learning framework can benefit from documents without annotations via a self-supervised training strategy. In the pre-training stage, we have a masking function $f_{\text{Mask}}(\cdot)$ that randomly mask selected proposals in a document in language or vision branch with a pre-defined probability that can 1) set the language or vision feature to zeros, or 2) replace the feature with a random proposal in the same modality from the pre-training corpus, or 3) keep the original feature unchanged. In the pre-training stage, we minimize the pre-training objective function as follows:

$$\begin{aligned} L =& \mathbb{E}_{\mathcal{D}^{\text{lang}}_{\text{Mask}}} L_1(x^i_{\text{lang}} - f_{\text{SelfDoc}}(f_{\text{Mask}}(x^i_{\text{lang}})|D)) \\ &+ \mathbb{E}_{\mathcal{D}^{\text{visn}}_{\text{Mask}}} L_1(x^i_{\text{visn}} - f_{\text{SelfDoc}}(f_{\text{Mask}}(x^i_{\text{visn}})|D)), \end{aligned} \tag{9}$$

where $\mathcal{D}^{\text{lang}}_{\text{Mask}}$ and $\mathcal{D}^{\text{visn}}_{\text{Mask}}$ are the distributions for masked proposals on language and vision branches, respectively. $f_{\text{SelfDoc}}(\cdot)$ denotes our whole model which outputs a feature embedding for each proposal. L_1 represents Smooth L1 loss [10]. D represents the document and, in this context, can be viewed as the surrounding features of proposals of the masked features from the two modalities.

The proposal selection and masking function are applied independently to language and vision features. The pre-training objective function working with modality interaction not only infers the masked features from surrounding proposals in the same modality, but can also absorb features from another modality and encourage cross-modal learning.

3.6. Modality-Adaptive Attention

In the fine-tuning phase and downstream usage, we fuse the output features for each proposal from both language and vision modalities. Most previous multimodal works [13, 36] use a simple linear additive operation for fusion. Considering the diverse variety of document images, we propose a modality-adaptive attention (M-AA) for a better feature fusion. The general idea is to apply sample-dependent attention weights to the two modalities and emphasize or diminish the intensity of language or vision features adaptively for different documents. Intuitively, this input-dependent attention can be helpful on some samples in raw documents such as: 1) ones that contain handwriting that is not recognizable by OCR algorithms, in which case a stronger emphasis on visual clue is needed; 2) documents that already contain abundant linguistic information such as scholarly articles, in which case a stronger emphasis on the semantic meaning of language is more helpful.

We summarize the pipeline for this module in Fig. 2(B). To be specific, we concatenate the output features of each proposal from language and vision branches, and feed it to a non-linear mapping network $\mathbb{R}^{2\times d_h} \rightarrow \mathbb{R}^2$ (d_h is the dimension of output features in either language or vision branch), then split the output weights into $w_{\text{lang}} \in \mathbb{R}^1$ and $w_{\text{visn}} \in \mathbb{R}^1$, and return the weights separately to its respective modality to perform element-wise product. We multiply the language and vision features with their modality-specific attention weight, then after a residual connection, features from two modalities are fused by a linear additive function. In our implementation, we employ a two-layer neural network that ends with a sigmoid activation function to achieve non-linear mapping.

4. Experiments

4.1. Implementation

Dataset. We use the PubLayNet dataset [38] and DocVQA dataset [23] to train the document object detector. PubLayNet includes 340K scholarly articles with bounding box on *text block*, *heading*, *figure*, *list*, and *table*, and DocVQA has 12K forms with a bounding box annotated for each text block. We use the official OCR results provided by the DocVQA website as the bounding boxes for text blocks to train the detector. We pre-train SelfDoc on the RVL-CDIP dataset [12], a document classification dataset containing 320K documents for training, 40K for validation, and 40K for testing. Pre-training is only conducted on the training set. Fig. 3 shows some image samples in document object detection, entity recognition, and document classification.

Document pre-processing. We train the document object detector using Detectron2 [34] with the ResNeXt-101 [35] backbone model. We apply rotations on images as data augmentation to improve the detection of the potential vertical text in documents. After obtaining the detection results, we use Tesseract OCR [16], a public OCR engine, to extract the plain text from each proposal given by detector. We crop the proposals from the original document images, and expand the bounding box by a factor of 1.1 and apply $2\times$ image magnification to better recognize the characters close to the edge and the overall word recognition. We convert detected OCR results into lower case, and convert digits to words. Common contractions are expanded before tokenization. For sentence embedding, we use the pre-trained sentence encoder (`bert-large-nli-mean-tokens`)[2]. For visual feature extraction, we concatenate the feature from the last and P2 layers (second to last convolutional layer) in RoI heads. We have $d_{\text{visn}} = 2048$ and $d_{\text{lang}} = 1024$.

[2]https://github.com/UKPLab/sentence-transformers

Table 1. Experimental results and comparison on document entity recognition in FUNSD dataset and document classification in RVL-CDIP dataset. The symbol ‡ implies feature fusing with global visual features on the whole document images from VGG-16. We re-implement LayoutLM on document classification and denote the result as 'our impl.', while the better results denote as ♯ is achieved using another data source. Please refer to the paragraph document classification in Sec. 4.2 for explanation on 'our impl.' and ♯.

Method	# Pre-training Data	Modality	Architecture	Entity Recognition	Classification
VGG-16	-	Vision	-	-	0.9031
ResNet-50	-	Vision	-	-	0.8866
Multimodal Ensemble [8]	-	Language + Vision	MLP + VGG-16	-	0.9303
Jain and Wigington [13]	-	Language + Vision	MLP + VGG-16	-	0.9360
Sentence-BERT [27]	-	Language	-	0.6947	-
$\text{BERT}_{\text{BASE}}$ [9]	-	Language	-	0.6062	0.8610
$\text{RoBERTa}_{\text{BASE}}$ [19]	-	Language	-	0.6648	0.8682
Pramanik *et al.* [25]	110K	Language + Vision + Layout	-	0.7744	0.9172
$\text{LayoutLM}_{\text{BASE}}$ [36]	500K	Language + Layout	-	0.6985	0.9125
$\text{LayoutLM}_{\text{BASE}}$ [36]	1M	Language + Layout	-	0.7299	0.9148
$\text{LayoutLM}_{\text{BASE}}$ [36]	2M	Language + Layout	-	0.7592	0.9165
$\text{LayoutLM}_{\text{BASE}}$ [36]	11M	Language + Layout	-	0.7866	0.9178
$\text{LayoutLM}_{\text{LARGE}}$ [36]	1M	Language + Layout	-	0.7585	0.9188
$\text{LayoutLM}_{\text{LARGE}}$ [36]	11M	Language + Layout	-	0.7789	0.9190
$\text{LayoutLM}_{\text{BASE}}$ [36]	1M	Language + Vision + Layout	-	0.7441	0.9431♯
$\text{LayoutLM}_{\text{BASE}}$ [36]	11M	Language + Vision + Layout	-	0.7927	0.9442♯
$\text{LayoutLM}_{\text{BASE}}$ (our impl.)	11M	Language + Layout	-	0.7887	0.8857
$\text{LayoutLM}_{\text{BASE}}$ (our impl.)	11M	Language + Vision + Layout	-	0.7993	0.9169‡
SelfDoc	Scratch	Language + Vision + Layout	w/o M-AA	0.7607	0.9049
SelfDoc	320K	Language + Vision + Layout	w/o M-AA	0.8263	0.9263/0.9364‡
SelfDoc	320K	Language + Vision + Layout	with M-AA	**0.8336**	0.9281/**0.9381**‡

Pre-training. Due to their variety, the number of proposals in documents may vary significantly. This variance could cause some input sequences to be heavily padded to ensure that all sequences are as long as the longest sequence in batch-wise training, therefore slowing down the training speed. To deal with this issue, we do not only set a maximum length of input sequences, but also apply batch thresholding to avoid excessive padding for some documents and reduce some batches in the sequence length when feasible. Every batch contains documents that have the number of proposals concurrently below or beyond the threshold. We set the maximum length of proposals to 50, and the group threshold to 30. When proposals exceed the maximum limitation, we randomly sample the proposals in this document. The input sentence sequence and RoI sequence are ended with the special token [SEP], respectively. For the model architecture, we assign 4 layers to each of the textual and visual encoders, and continue with 2 cross-modality layers. The total number of layers for each branch is 12 and is equivalent to $\text{BERT}_{\text{BASE}}$. We keep other specific architectural configurations and masking probability the same as in $\text{BERT}_{\text{BASE}}$. In the pre-training phase, we set the batch size to 768, learning rate to 1e-4 in AdamW optimizer [20] with a linear warm-up ratio to 0.05 and linear decay. We do not initialize our model before pre-training with parameters from pre-trained BERT or any variants. We conduct pre-training for 72K iterations on 8 Tesla V-100 GPUs and it takes around 21 hours to complete the pre-training.

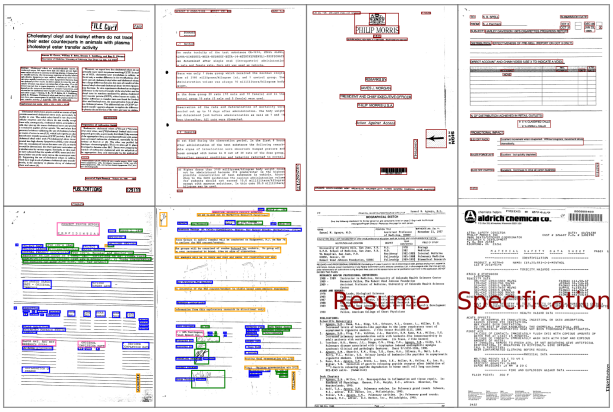

Figure 3. Example outputs. First row: examples of document object detection. Second row: examples of document entity recognition with different colors indicating different entity categories (the left two), and document classification with labels (the right two).

4.2. Applications

Document entity recognition. The first downstream task for evaluation is document entity recognition. We adopt the FUNSD dataset [14] instead of SROIE [1] since FUNSD is an in-domain dataset. It contains 149 forms with 7,441 en-

Table 2. Experimental results on document clustering over different number of cluster centroids and samples.

Method	# Cluster = 4		# Cluster = 6		# Cluster = 8		# Cluster = 10		# Cluster = 12	
	Acc.	NMI	Acc.	NMI	Acc.	NMI	Acc.	NMI	Acc.	NMI
$\text{BERT}_{\text{BASE}}$ [9]	0.4190	0.1589	0.3743	0.1568	0.3400	0.1670	0.3903	0.2752	0.3021	0.2378
$\text{LayoutLM}_{\text{BASE}}$ [36]	0.4380	0.2107	0.4147	0.2111	0.3612	0.1821	0.3237	0.2213	0.2646	0.1979
Input Embedding	0.4700	0.1946	0.3937	0.1723	0.3898	0.2636	0.3457	0.2549	0.3033	0.2315
SelfDoc	**0.6970**	**0.4155**	**0.4550**	**0.3025**	**0.4476**	**0.3473**	**0.4010**	**0.3612**	**0.3614**	**0.3173**

tities for fine-tuning, and 50 forms with 2,332 entities for testing. Semantic entities are classified into four categories: '*header*', '*question*', '*answer*', or '*other*'. The bounding box and plain text are given for each block of text as well for every single word, but we only make use of the positional and textual information for text blocks. We extract visual features from the RoI head in detector using the given box coordinates for each block, and extract the language embedding using the plain text as described previously. Micro averaged F1 score is used as the evaluative metric. We train a linear classifier on the output of our pre-trained model in the fine-tuning phase, with parameters in the pre-trained model fixed. The fine-tuning phase takes 60 epochs with a learning rate of 5e-5 and batch size of 16. Note that the experiment of FUNSD in LayoutLM [36] considers the prediction of word positions (begin, intermediate, end). Since our model considers the whole block of text, the word position is obvious and results remain the same in that setting.

Document classification. We use the RVL-CDIP [12] dataset to evaluate the performance of document classification, using the training set for fine-tuning and the testing set for evaluation. It consists of 400k images in 16 classes. We take the encoder outputs on the special tokens [LANG] and [VISN] from the modality-adaptive attention module as holistic representations of the textual and visual inputs. The addition of two features is used as the input to the classifier. The whole fine-tuning takes 20K iterations with a batch size of 768 and a learning rate of 5e-5.

Several technical issues related to the baseline model LayoutLM [36] in this task, drive us to re-implement this model to make a fair comparison. 1) In their experiments, LayoutLM does not use document images from RVL-CDIP, instead, they retrieve the corresponding images from IIT-CDIP test collection 1.0 [17], which is a superset of RVL-CDIP but contains high-quality document images. The most obvious advantage is on the detected results of OCR, where the text is cleaner and more informative. Unfortunately, we had issues [2] accessing IIT-CDIP; 2) LayoutLM [36] uses an image embedding from a detector over the whole document image on this task, and jointly trains the detector during fine-tuning. However, the released code does not contain the jointly fine-tuned detector model. Given these two technical issues, we built our implementation using the released pre-trained models $\text{LayoutLM}_{\text{BASE}}$ and fine-tuning protocols to make the result a fair comparison, and denote it as 'our impl.'. We also tried to fine-tune the released $\text{LayoutLM}_{\text{LARGE}}$ model but faced a convergence issue due to the need to restrict the batch size for computational reasons.

We hereby provide our model and LayoutLM with the same source of data, and also provide the results fusing with the same image embedding from VGG-16 [30] that trained on RVL-CDIP. We choose the embedding from VGG-16 instead of other sophisticated models since some previous work [8, 13] use VGG-16 on document classification.

Document clustering. We also investigate models in the scenario where there is no annotation available. We consider document clustering on RVL-CDIP [12] testing set. We randomly sample from the testing set and create five experimental scenarios, with {3, 5, 7, 10, 12} clusters. The corresponding numbers of samples are {1k, 3k, 5k, 7k, 9k}. In this task, we also include the input embedding (sentence embedding and visual features with the layout) of our model for comparison. Since all models do not have a supervision or pre-training objective function at the document image level, we take the average of input proposal sequence or word sequence as a representation of the document, and conduct K-means [3] clustering over all the document representations. The model for pre-trained $\text{LayoutLM}_{\text{BASE}}$ is directly used. We use metrics clustering accuracy (Acc.) and normalized mutual information (NMI) for evaluation.

4.3. Result and Discussion

Quantitative results are listed in Table 1 & 2, and an ablation study on SelfDoc is presented in Table 3. We discuss our observations from the experiments as follows.

Baselines. We include five task-specific baselines in Table 1. These include two standard convolutional neural networks VGG-16 and ResNet-50, two multimodal ensemble approaches [13, 8] using VGG-16 and a neural network for text encoding, plus the Sentence-BERT embedding. We have four task-agnostic learning methods, including two pre-trained language models [9, 19], the approach proposed by Pramanik *et al.* [25] pre-trained on arXiv dataset [4], and LayoutLM [36] pre-trained on IIT-CDIP dataset [17].

SelfDoc outperforms baselines. Table 1 & 2 show that SelfDoc outperforms baselines on document entity recognition, document classification and clustering. The only result

Table 3. Ablation studies on SelfDoc on multimodal information, pre-training data, and cross-modality attention.

Setting	Modality	Parameter	Entity Recognition	Classification
Scratch, remove layout	Language + Vision	137M	0.7579	0.9070
Scratch, remove language	Vision + Layout	60M	0.6209	0.8447
Scratch, remove vision	Language + Layout	60M	0.7491	0.8895
Pre-train with 40K data	Language + Vision + Layout	137M	0.7886	0.9119
Pre-trained w/o CrossAtt-1&2	Language + Vision + Layout	146M	0.7911	0.9189
Pre-trained w/o CrossAtt-1	Language + Vision + Layout	137M	0.8152	0.9224
Pre-trained w/o CrossAtt-2	Language + Vision + Layout	137M	0.8130	0.9229

we cannot outperform is the reported number by LayoutLM which uses a cleaner data source and jointly fine-tunes with a deeper CNN model, as we discussed in Sec. 4.2. Other than that, SelfDoc demonstrates a good performance on all three evaluative downstream tasks with significantly fewer data for pre-training. We deem the effective usage of pre-training documents as a superior advantage of our model since documents often contain sensitive information, thus the legal and privacy issues may limit the feasibility to build large scale high-quality document datasets for pre-training.

A proposal is richer than a single word. From the document clustering results in Table 2, we observe that our input embedding is able to deliver a more informative representation than BERT and LayoutLM. This indicates that our pre-trained model can further improve the discriminability in features without fine-tuning. Experiments on clustering suggest that on the representation of the whole document image, exploring information from proposals can be more helpful than collecting features from each word. In addition, well-designed modeling on feature embedding can also bring informative representation.

Multimodal modeling is beneficial. In the first three rows of our ablation study in Table 3, we consider the scenario where removal happens on the three input components of SelfDoc: textual input, visual input, and structural layout, one at a time. We observe a significant drop in performance when removing textual or visual input on both entity recognition and classification, confirming the necessity for learning on two modalities. The structural layout has a smaller effect but it still contributes to the classification.

Effectiveness of cross-modal learning. In the last three rows of Table 3, we investigate the effectiveness of cross-modal learning in our model. Note that we do not shrink the number of model parameters when removing a cross-modality attention function. When removing a single attention function (denoted as w/o CrossAtt-1/2), we fill the space with the remaining attention function. When totally removing the cross-attention encoder, we deepen the single modality encoder to maintain the size of the model. The results demonstrate the importance of both cross-modal learning and the mixture of two cross-attention functions.

A complementary global visual feature is helpful. A feature embedding from convolutional neural networks on the whole image can improve our result by around 1% in document classification. The improvement comes from 1) sometimes document object detector gives an empty detection on low-quality pages, making an all-zero input for our model, so the model learns to exploit global visual feature, and 2) some texts in the proposals are obscured, thus OCR might deliver a random result, making the language embedding uninformative, so an external feature is beneficial.

Table 4. Fine-tuning with fewer data in document classification.

	w/o VGG-16	with VGG-16
$\text{LayoutLM}_{\text{BASE}}$ (our impl.)	0.8544	0.8712
SelfDoc	0.8929	0.9150

Fewer data for fine-tuning. We also provide a scenario where fewer available labels can be accessed in document classification in Table 4. To do so, we fine-tune our model and LayoutLM on the validation set of the RVL-CDIP dataset, resulting in an 8 times reduction of fine-tuning data. The VGG-16 model used for fusing is also trained with fewer data. The quantitative results show that our model surpasses LayoutLM by a larger margin compared to fine-tuning with much more data. The observation confirms that our model is also effective when fine-tuning labels are rare.

5. Conclusion

We proposed a task-agnostic framework for representation learning and pre-training on document images. Our framework was defined at the semantic components level (rather than words), fully considers the presented property of document data, and includes linguistic, visual, and structural layout information. We employed contextualized learning on the sequential proposals, and encouraged cross-modal learning across language and vision by the proposed cross-modal encoder. We used modality-adaptive attention to emphasize features in language and vision for multi-modal fusion. With significantly fewer data for pre-training, we achieved superior performance on multiple tasks.

Acknowledgments

This work was supported in part by Adobe Research and NSF OAC 1920147.

References

[1] Icdar 2019 robust reading challenge on scanned receipts ocr and information extraction. https://rrc.cvc.uab.es/?ch=13&com=introduction.

[2] Iit-cdip test collection is unavailable? https://github.com/microsoft/unilm/issues/250, 2020.

[3] David Arthur and Sergei Vassilvitskii. k-means++: The advantages of careful seeding. Technical report, Stanford, 2006.

[4] Arxiv. arxiv bulk data access. https://arxiv.org/help/bulk_data, 2020.

[5] Jimmy Lei Ba, Jamie Ryan Kiros, and Geoffrey E Hinton. Layer normalization. *arXiv preprint arXiv:1607.06450*, 2016.

[6] Iz Beltagy, Matthew E Peters, and Arman Cohan. Longformer: The long-document transformer. *arXiv preprint arXiv:2004.05150*, 2020.

[7] Adrian Cosma, Mihai Ghidoveanu, Michael Panaitescu-Liess, and Marius Popescu. Self-supervised representation learning on document images. In *DAS*, 2020.

[8] Tyler Dauphinee, Nikunj Patel, and Mohammad Rashidi. Modular multimodal architecture for document classification. *arXiv preprint arXiv:1912.04376*, 2019.

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

[10] Ross Girshick. Fast r-cnn. In *CVPR*, 2015.

[11] Jiuxiang Gu, Jason Kuen, Shafiq Joty, Jianfei Cai, Vlad Morariu, Handong Zhao, and Tong Sun. Self-supervised relationship probing. In *NeurIPS*, 2020.

[12] Adam W Harley, Alex Ufkes, and Konstantinos G Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *ICDAR*, 2015.

[13] Rajiv Jain and Curtis Wigington. Multimodal document image classification. In *ICDAR*, 2019.

[14] G. Jaume, H. Kemal Ekenel, and J. Thiran. Funsd: A dataset for form understanding in noisy scanned documents. In *ICDARW*, 2019.

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

[16] Anthony Kay. Tesseract: An open-source optical character recognition engine. *Linux J.*, 2007(159):2, July 2007.

[17] D. Lewis, G. Agam, S. Argamon, O. Frieder, D. Grossman, and J. Heard. Building a test collection for complex document information processing. In *SIGIR*, 2006.

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

[19] Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *CoRR*, abs/1907.11692, 2019.

[20] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *ICLR*, 2018.

[21] Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. In *NeurIPS*, 2019.

[22] Simone Marinai, Marco Gori, and Giovanni Soda. Artificial neural networks for document analysis and recognition. *TPAMI*, 2005.

[23] Minesh Mathew, Dimosthenis Karatzas, R Manmatha, and CV Jawahar. Docvqa: A dataset for vqa on document images. In *WACV*, 2021.

[24] Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. Deep contextualized word representations. In *NAACL*, 2018.

[25] Subhojeet Pramanik, Shashank Mujumdar, and Hima Patel. Towards a multi-modal, multi-task learning based pre-training framework for document representation learning. *arXiv preprint arXiv:2009.14457*, 2020.

[26] Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. Improving language understanding by generative pre-training. https://s3-us-west-2.amazonaws.com/openai-assets/research-covers/language-unsupervised/language_understanding_paper.pdf, 2018.

[27] Nils Reimers and Iryna Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. In *EMNLP*, 2019.

[28] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In *NeurIPS*, 2015.

[29] Michael Shilman, Percy Liang, and Paul Viola. Learning nongenerative grammatical models for document analysis. In *ICCV*, 2005.

[30] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015.

[31] Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. In *EMNLP*, 2019.

[32] 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.

[33] Hao Wei, Micheal Baechler, Fouad Slimane, and Rolf Ingold. Evaluation of svm, mlp and gmm classifiers for layout analysis of historical documents. In *ICDAR*, 2013.

[34] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019.

[35] Saining Xie, Ross Girshick, Piotr Dollár, Zhuowen Tu, and Kaiming He. Aggregated residual transformations for deep neural networks. In *CVPR*, 2017.

[36] 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 *SIGKDD*, 2020.

[37] Xiao Yang, Ersin Yumer, Paul Asente, Mike Kraley, Daniel Kifer, and C Lee Giles. Learning to extract semantic structure from documents using multimodal fully convolutional neural networks. In *CVPR*, 2017.

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

A. Visualization of Modality-Adaptive Attention

We visualize the attention mechanism in Fig. 4. The provided samples show that it gives attention scores for each modality adaptively on different types of documents. Some documents with heavy-word or fewer clues in vision have a larger value in w_{lang}, while some forms with multiple font styles or unrecognizable hand-written enjoy a larger value in w_{visn}.

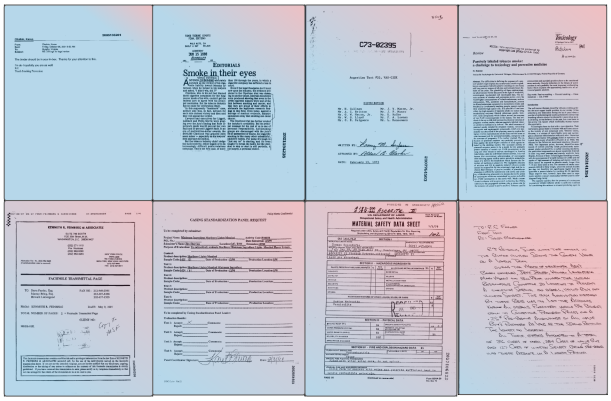

Figure 4. Visualization of Modality-Adaptive Attention on document classification. The size of covering area in blue and red represents the value in w_{lang} and w_{visn}, respectively.

B. Experiments in Document Clustering

B.1. Evaluative metrics

$$\text{Acc.} = \frac{\sum_{i=1}^{n} \mathbb{1}_{y_i = \text{map}(\hat{y}_i)}}{n},$$

$$\text{NMI} = \frac{\sum_{i,j} n_{ij} \log \frac{n \cdot n_{ij}}{n_{i+} \cdot n_{+j}}}{\sqrt{(\sum_i n_{i+} \log \frac{n_{i+}}{n})(\sum_j n_{j+} \log \frac{n_{+j}}{n})}},$$

where $\mathbb{1}$ is the indicator function, and $\text{map}(\hat{y}_i)$ is permutation mapping function that maps each cluster label $\hat{y}_i$ to the ground truth label y_i using linear sum assignment, n_{ij}, n_{i+} and n_{+j} represent the co-occurrence number and cluster size of i-th and j-th clusters in the obtained partition and ground truth, respectively, and n is the total data instance number.

B.2. Label sets

Cluster = 4: {email, form, handwritten, letter}
Cluster = 6: {email, form, handwritten, letter, news article, resume}
Cluster = 8: {email, form, handwritten, letter, news article, questionnaire, resume, scientific publication}
Cluster = 10: {email, file folder, form, handwritten, letter, news article, questionnaire, resume, scientific publication, specification}
Cluster = 12: {email, file folder, form, handwritten, letter, memo, news article, questionnaire, resume, scientific publication, scientific report, specification}