DocTr: Document Transformer for Structured Information Extraction in Documents

Haofu Liao[1*] Aruni RoyChowdhury[2†] Weijian Li[3†] Ankan Bansal[1] Yuting Zhang[1]
Zhuowen Tu[1] Ravi Kumar Satzoda[1] R. Manmatha[1] Vijay Mahadevan[1]

[1]AWS AI Labs [2]MathWorks [3]Amazon Physical Stores

Abstract

We present a new formulation for structured information extraction (SIE) from visually rich documents. We address the limitations of existing IOB tagging and graph-based formulations, which are either overly reliant on the correct ordering of input text or struggle with decoding a complex graph. Instead, motivated by anchor-based object detectors in computer vision, we represent an entity as an anchor word and a bounding box, and represent entity linking as the association between anchor words. This is more robust to text ordering, and maintains a compact graph for entity linking. The formulation motivates us to introduce 1) a Document Transformer (DocTr) that aims at detecting and associating entity bounding boxes in visually rich documents, and 2) a simple pre-training strategy that helps learn entity detection in the context of language. Evaluations on three SIE benchmarks show the effectiveness of the proposed formulation, and the overall approach outperforms existing solutions.

1. Introduction

Structured information extraction (SIE) from documents, as shown in Fig 1, is the process of extracting entities and their relationships, and returning them in a structured format. Structured information in a document is usually *visually-rich* – it is not only determined by the content of text but also the layout, typesetting, and/or figures and tables present in the document. Therefore, unlike the traditional information extraction task in nature language processing (NLP) [8, 3, 30] where the input is plain text (usually with a given reading order), SIE assumes the image representation of a document is available, and a pre-built optical character recognition (OCR) system may provide the unstructured text (i.e., without proper reading order). This is a practical assumption for day-to-day processing of business documents, where the documents are usually stored as images or PDFs, and the structured information, such as key-value pairs or line items (see Fig. 2) from invoices and receipts, has been primarily obtained manually. This is time consuming and does not scale well. Hence, automating the document structured information extraction process with efficiency and accuracy is of great practical and scientific importance.

Structured information extraction is part of *document intelligence* [5], which focuses on the automatic reading, understanding, and analysis of documents. Early approaches to document intelligence usually address the problem purely

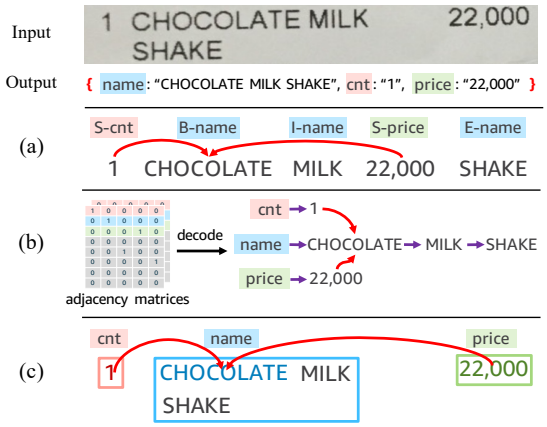


Figure 1: Structured information extraction problem formulations. Given an input image, we aim to extract each entity (e.g., `name`, `count`, or `price`) and link the related entities together. To address this task, **(a) IOB tagging** [29] assigns a tag to each word to indicate if it is the beginning (B-), inside (I-), end (E-) of an entity or a single (S-) word entity. **(b) Graph based methods** [15] take each word as a node, and use edges between words to indicate that the words belong to the same entity (purple edges) or the underlying entities are linked (red edges). A graph is generated by decoding from two adjacency matrices (one for each type of edges). **(c) Our formulation** represents an entity as an anchor word (colored words) and a box (colored bounding boxes), and represents entity linking via anchor word association (red arrows).

*Corresponding author liahaofu@amazon.com
†Work done at AWS AI Labs

from either a computer vision or an NLP perspective. The former takes the document as an image input and frames entity detection as object detection or instance segmentation [41, 31]. The latter takes only the textual content of a document as the input, and addresses the problem with NLP solutions, such as IOB tagging via transformers [14].

Recently, models have also been proposed to pre-train on large-scale document collections and apply them to a wide variety of downstream document intelligence problems [38, 11, 1, 20]. Such general-purpose models usually have the ability to make use of multi-modal inputs – text from OCR, layout in the form of text locations, and visual features from images, and pre-training enables them to understand the basic structure of documents. Therefore, general-purpose models have demonstrated significant improvements on multiple document intelligence tasks, such as entity extraction [11, 20], document image classification [38, 1], and document visual question and answering [39, 1].

For structured information extraction, existing general-purpose models rely on two broad approaches: 1) IOB tagging [29] based methods [38, 39, 20], and 2) graph based methods [11, 15]. Both of these approaches suffer from inherent limitations. IOB tagging relies on the correct "reading order" or serialization of text, which however is not given by the OCR. As shown in Fig. 1(a), the raster scan order of OCR text separates `I-name` and `E-name`. When there are multiple `name` entities, it could be non-trivial to know which `I-name`/`E-name` word belongs to which `name` entity. Graph-based methods (Fig. 1 (b)) can result in complex graphs with many words in a document (i.e., many nodes in the graph). Therefore, decoding the entities and their relationships from the adjacency matrices is error-prone.

Given the limitations of existing work, we make the following contributions in this paper:

- We introduce *a new formulation* for SIE where we represent an entity as an anchor word along with a box, and regard the problem as an anchor word based entity detection and association problem (Fig 1 (c)). Thus, we extract entities via bounding boxes and do not depend on the reading order of input. We assign each entity with an anchor word, resulting in a compact graph of entity relations (e.g., the anchor word links in Fig 1 (c)), which facilitates decoding structured information.
- We develop *a new model*, called Document Transformer (DocTr), which combines a language model and visual object detector for joint vision-language document understanding. We note that the recognition of an anchor word is largely a language-dependent task, while the detection of entity boxes is a more vision-dependent task. Therefore, DocTr is an intuitive approach to target this problem under the proposed formulation.
- We propose *a new pre-training task*, called masked detection modeling (MDM), that matches our formulation and helps learn box prediction in the context of language. Our experimental results show that 1) the proposed formulation addresses SIE better than IOB tagging or graph-based solutions, 2) MDM is a more effective pre-training task, in particular when worked together with the new formulation, and 3) the overall approach outperforms existing solutions on three SIE tasks.

2. Related Work

General-purpose document understanding. General-purpose approaches aim to develop a backbone model for document understanding, which is then adapted to address downstream document understanding tasks. LayoutLM [38, 39, 13] is an early approach that pre-trains on a large-scale document dataset. It introduces masked vision-language modeling and layout information for document understanding pre-training. BROS [11] improves LayoutLM via better encoding of the spatial information and introducing a pre-training loss for understanding text blocks in 2D. DocFormer [1] introduces a new architecture and pre-training losses to better leverage text, vision and spatial information in an end-to-end fashion. FormNet [20] encodes neighborhood context for each token using graph convolutions and introduces an attention mechanism to address imperfect serialization. StrucText [23] proposes to extract multi-modal semantic features at both token level, word-segment level and/or entity level. Donut [19] proposes an OCR free solution that is pre-trained to predict document text from images. It is an encoder-decoder model that can directly decode the expected outputs as text for downstream tasks.

Structured information extraction (SIE). Early approaches [41, 18, 6] formulate the SIE problem as a computer vision problem to either segment or detect entities from documents. However, they cannot address linking of entities due to the limitation of the formulation. With the advent of transformers [34] and their success in NLP, more recent approaches [25, 44, 10] address SIE by incorporating layout/visual information with text inputs to transformers, and extract entities via a NLP formulation [29]. Other approaches [24, 36, 42] propose to regard the text inputs as the nodes in a graph and model the relationship of text inputs via graph neural networks. To extract the relationship between entities, SPADE [15] introduces a graph decoding scheme on learned pairwise affinities between extracted entities.

Table detection and recognition (TDR). TDR is the task of detecting and recognizing tabular structures from document images. Both SIE and TDR focus on returning information in a structured way from documents. However, unlike SIE where the spatial relationship of entities are unconstrained, TDR assumes a tabular structure of entities (i.e., table cells) and leverages this prior knowledge in the model design and post-processing [28, 45, 26]. Moreover,

SIE requires returning a semantic label for each entity which demands an understanding of the text, while TDR does not distinguish between types of table cells but focuses more on table layout. Therefore, the existing approaches [28, 45, 26] to TDR are vision-only approaches.

TextVQA. Given an input image, TextVQA aims to answer questions related to the text in image. Similar to SIE, existing TextVQA approaches [32, 12, 9, 2] employ multi-modal models that take both the OCR and image as inputs. However, for TextVQA, the answers are typically single entities. It can be challenging to address the problem with TextVQA if we aim to return multiple entities in a structured way, and if an image could have multiple of such structures.

Scene graph generation (SGG). Generating scene graphs can be regarded as a form of SIE for natural images. SGG methods [17, 37, 43, 40, 33] detect objects as the nodes of scene graphs, and construct edges of scene graphs by identifying the pairwise relationships between objects. This is similar to our formulation of SIE where we extract entities via anchor word guided object detection, and link entities by learning to output their pairwise affinities.

3. Approach

3.1. Structured Information Extraction

Problem Formulation. Following prior work, we assume the input is the image of a document page, and a pre-built OCR system is applied to detect and recognize the words. The goal of a structured information extraction system for document understanding is to extract a set of grouped entities $\mathcal{G} = \{\mathbf{G}_i\}$, where each entity group $\mathbf{G}_i = \{\mathbf{e}_{ij}\}$ is a set of entities with predefined relations. As shown in Fig. 2, an entity group may be a key and value pair, or a line item containing the name, count and price entities. We denote an entity as $\mathbf{e} = (t, c, b)$ where t, c and b are the text, class label, and location (bounding box) of the entity, respectively. Note that, with OCR inputs, this formulation of an entity can be reduced to $\mathbf{e} = (c, b)$, because the text t can be obtained by aggregating the OCR text inside b.

Next, we propose a *new formulation* to address structured information extraction. We propose to address *entity extraction* via anchor word guided detection and *entity linking* via anchor word association. The former extracts entities $\{\mathbf{e}_i\}$, and the latter links entities into groups $\{\mathbf{G}_i\}$.

Entity Extraction via Anchor Word Guided Detection. To extract an entity $\mathbf{e}$, we first introduce a new concept called *anchor word*, which is a designated word of an entity. In Fig. 2, we select the first word of an entity as the anchor word, e.g., "ABC" is the anchor word for `value`, and "Chicken" is the anchor word for `name`. Other designations of anchor words are possible (see Sec. 4.2). An anchor word may be regarded as the representation of an entity. Since the goal of extracting an entity $\mathbf{e} = (c, b)$ is to find its class label c and bounding box b, they may then be represented by an anchor word. As shown in Fig. 2, we associate each anchor with a label and a bounding box. For example, the anchor word "Ship" is associated with a label `key` and a bounding box that encloses the entity "Ship To:". Therefore, the task of extracting an entity may be seen as first identifying its anchor word, and then obtaining the label and bounding box associated with it.

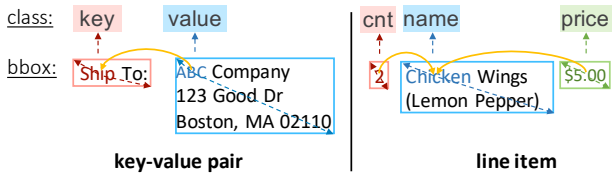


Figure 2: Illustration of our formulation for structured information extraction of two types of relations. We first identify the "anchor words" of entities (which are the first words of entities in this example, e.g., "Ship" or "Chicken"). Then, from each anchor word, we extract the entity by predicting (dotted arrows) its class label and bounding box. We link entities by linking (yellow arrows) their anchor words. For key-value pair relation, we link "Ship" to "ABC". For line item relation, we link "2" and "$5.00" to "Chicken".

Entity Linking via Anchor Word Association. We define an entity group as consisting of a *primary* entity, and all the other entities in the group are *secondary*. The anchor word of a primary/secondary entity is the primary/secondary anchor word. Once anchor words have been identified, linking entities into entity groups is equivalent to associating anchor words. To establish such association, we first select the primary anchor words of entity groups, and then all the secondary anchor words from the same group are linked to the primary anchor word. The definition of a primary entity may vary. For key-value pairs, the primary anchor words may simply be those anchor words labeled as `key`. For more general entity groups, we designate a primary anchor word based on the task/data. For example, we choose `name`'s anchor word "Chicken" as the primary anchor in Fig. 2. Other ways of choosing primary anchors are possible (See Sec. 4.2). Links between primary and secondary anchor words are represented by a binary matrix $\mathbf{M} \in \{0, 1\}^{m \times n}$. $\mathbf{M}_{ij} = 1$ indicates that the ith primary anchor word, and jth secondary anchor word are linked. Otherwise, $\mathbf{M}_{ij} = 0$.

3.2. DocTr: Document Transformer

DocTr is a multi-modal transformer that takes both the document image and OCR words (text and position) as input. Unlike existing encoder-only approaches [39, 1, 23], DocTr has an encoder-decoder architecture with 1) two dedicated encoders to encode vision and language features separately,

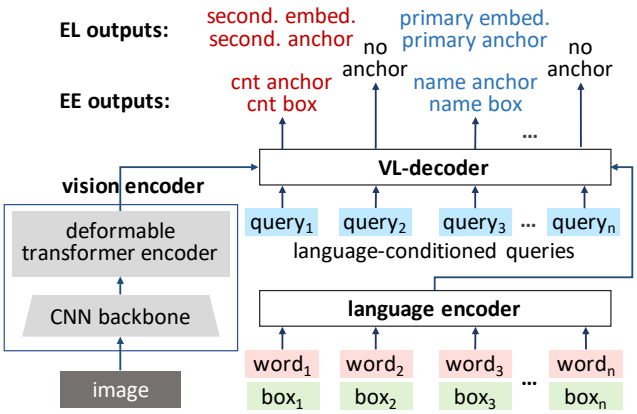


Figure 3: Overall architecture of DocTr. The vision encoder extracts visual features from a document image. The language encoder extracts language features from OCR text and bounding boxes (i.e., document layout information). The VL-decoder uses language-conditioned queries to decode structured information from visual and language features. For entity extraction (EE), each query decodes an anchor word label and an entity box. For entity linking (EL), each query decodes an association embedding of an anchor word and a primary/secondary anchor label.

and 2) a vision-language decoder to decode anchor word based outputs for entity extraction and entity linking. An overview of the DocTr architecture is shown in Fig. 3.

Vision Encoder. The vision encoder is adapted from Deformable DETR [46]. It consists of a CNN backbone with multi-scale visual feature extraction, and a deformable transformer encoder for efficient encoding of visual features. Compared with vanilla transformer based vision encoders, this design is more lightweight due to the use of deformable attention, which has linear complexity with respect to the spatial size of image feature maps instead of the quadratic complexity using standard self-attention. As a result, it is capable of encoding high-resolution multi-scale visual features for better detection of small objects/entities.

This vision encoder is shown to work effectively with a transformer decoder for end-to-end object detection [4]. This is helpful to our formulation of entity extraction, where we convert this task into an anchor word guided object detection problem. We also highlight the differences from existing encoder-only methods where the visual features – either region-based [38, 23] or grid-based [39, 1] – are extracted with a pre-trained CNN model; they are sent to the transformer encoder along with OCR inputs without dedicated network components for decoding entity bounding boxes.

Language Encoder. The language encoder is a transformer model adapted from the BERT architecture [7]. We follow LayoutLM [38] to include the layout information (i.e., 2D position embeddings of OCR) along with the OCR text as input. However, no visual information is added since it has already been addressed by the vision encoder. The language encoder is critical to our formulation for the identification of anchor words, which is a language-dependent task.

Vision-Language Decoder with Language-Conditioned Queries. The architecture of the vision-language decoder is similar to the decoder of the Deformable DETR transformer model [46] - with two major differences to facilitate the decoding of vision-language inputs. Each decoder layer has two cross-attention modules to decode from vision and language inputs respectively. For vision, we apply *deformable cross-attention* (similar to Deformable DETR) to efficiently decode from high-resolution visual features. For language, we apply *language-conditioned cross-attention* to decode from the discrete OCR language features.

Specifically, we introduce *language-conditioned queries* to better leverage the OCR inputs and obviate the need for bipartite matching between predicted and ground truth entities. The original DETR-like decoder queries [4, 46] do not have explicit semantic meanings at the beginning. Hence, DETR requires finding the most plausible matching between a prediction and ground truth, which is less effective and impedes the training. For document understanding with OCR inputs, we consider a one-to-one mapping between OCR inputs and decoder queries. That is, we have the same number of queries as the number of OCR inputs to the language encoder, and the ith query is mapped to the ith OCR input (see Fig. 3). This mapping can be simply modeled as cross-attention between queries and language embeddings by using the same position embedding for both inputs. Let $\mathbf{Q} \in \mathbb{R}^{L\times d}$ be a set of L decoder queries each with dimension d (packed as a matrix), $\mathbf{V} \in \mathbb{R}^{L\times d}$ be the set of output embeddings from the language encoder, and $\mathbf{P} \in \mathbb{R}^{L\times d}$ be a set of position embeddings. Then, the cross attention with language-conditioned queries can be written as:

$$\text{CrossAttn}(\mathbf{Q}, \mathbf{V}, \mathbf{P}) = \text{softmax}(\frac{(\mathbf{Q}+\mathbf{P})(\mathbf{V}+\mathbf{P})^T}{\sqrt{d}})\mathbf{V}, \quad (1)$$

where $\sqrt{d}$ is a scaling factor [34]. This mapping assigns each query with an explicit linguistic semantic meaning – the i-th decoder output now corresponds to the i-th input text token, via the i-th decoder query. Thus, we can directly match entities with queries without the bipartite matching required by the default DETR decoder formulation [4, 46].

Entity Extraction and Linking Outputs. The decoder has two sets of outputs for entity extraction and entity linking respectively (see Fig. 3). For entity extraction, each output is a class label and a bounding box which uniquely decide an entity. Because each query (and thus its corresponding output) is mapped to an OCR input, the class label indicates

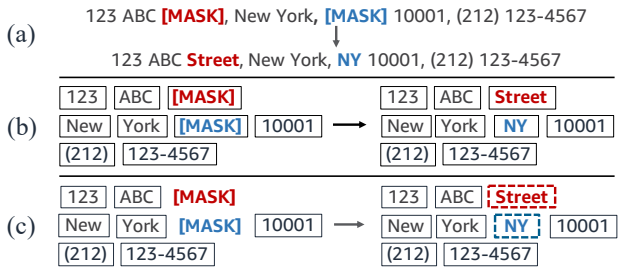


Figure 4: Illustration of of masked detection modeling (MDM) and comparison with masked language modeling (MLM) [7] and masked vision-language modeling (MVLM) [38]. **(a) MLM** takes only text as input, and requires predicting the masked text input. **(b) MVLM** takes both OCR texts and boxes as input. But only the text part is masked and to be predicted. **(c) MDM (ours)** takes one step further by masking both the texts and boxes, and requires predicting the masked boxes and their corresponding texts.

whether the underlying OCR input is an anchor word, and the type of entity it represents. For entity linking, each output is a binary class label and an embedding vector. The binary class label indicates whether the OCR input is a primary anchor word. The embedding vector is for the linking of anchor words, and we use different embeddings for primary and secondary anchor words. Let $\mathbf{E}_\text{p} \in \mathbb{R}^{m\times h}$ be a set of m primary embeddings, and $\mathbf{E}_\text{s} \in \mathbb{R}^{n\times h}$ be a set of n secondary embeddings, the predicted affinity matrix for entity linking is computed as $\hat{\mathbf{M}} = \text{sigmoid}(\mathbf{E}_\text{p}\mathbf{E}_\text{s}^T), \hat{\mathbf{M}} \in (0,1)^{m\times n}$.

3.3. Architecture Details

For the vision encoder, we use a ResNet50 backbone and a 6-layer deformable transformer encoder [46]. The backbone is initialized with ImageNet pretrained weights, and outputs three scales of visual features. The multi-scale visual features are transformed into a sequence with 2D "sine" position embeddings before sending to the deformable transformer encoder. For the language encoder, we use a 12-layer transformer encoder with the same architecture settings as the BERT-base model [7]. In addition to BERT's text embeddings and 1D position embeddings, we also add 2D position embeddings [38] to include layout information of the document as the input. The 2D position embeddings are learned embeddings with random initialization. The VL-decoder has 6 layers, where each layer consists of a self-attention module, a deformable cross-attention module [46] and a standard cross-attention module [34] (see supplementary material for detailed architecture of VL-decoder layers).

3.4. Training and Pre-training

Entity Extraction and Linking Objectives. The entity extraction objective is similar to the one used in DETR [4] except that we do not need the bipartite matching due to the use of language-conditioned queries (as introduced in Sec. 3.2). Specifically, given a set of N OCR inputs, the language-conditioned queries yields N entity extraction outputs $\hat{\mathbf{E}} = \{\hat{\mathbf{e}}_i\}_{i=1}^N$. For a document with M entities, we also construct a ground truth $\mathbf{E} = \{\mathbf{e}_i\}_{i=1}^N$ of size N. Here, $\hat{\mathbf{e}}_i$ and $\mathbf{e}_i$ denote the predicted and ground truth entities of the ith OCR input, respectively. Note that not every OCR word is an anchor word, and thus it may have no associated entity. In this case, we say that the ground truth of the input OCR is an empty entity, i.e., $\mathbf{e} = \varnothing$, and there are in total $N - M$ empty entities in $\mathbf{E}$. If we denote a non-empty entity as $\mathbf{e} = (c, b)$ and a predicted entity as $\hat{\mathbf{e}} = (\hat{p}, \hat{b})$, where c is the ground truth entity label, $\hat{p}$ is the predicted entity label probability, and b/$\hat{b}$ is the ground truth/predicted bounding box, then we write the entity extraction loss as

$$\mathcal{L}_\text{EE}(\mathbf{E}, \hat{\mathbf{E}}) = \textstyle\sum_i [-\log \hat{p}_i(c_i) + \lambda \mathbb{1}_{\{\mathbf{e}_i \neq \varnothing\}} \mathcal{L}_\text{bbox}(b_i, \hat{b}_i)], \tag{2}$$

where $\hat{p}_i(c_i)$ is the predicted probability of entity being labeled as c_i, $\mathcal{L}_\text{bbox}$ is a bounding box loss [4], and $\mathbb{1}_{\{\mathbf{e}_i \neq \varnothing\}}$ means we only compute $\mathcal{L}_\text{bbox}$ for non-empty entities.

The entity linking loss consists of two parts, primary anchor classification and linking classification. Let $\hat{\mathbf{L}}$ be a set of primary anchor classification outputs and $\mathbf{L}$ be its binary ground truth labels. Let $\hat{\mathbf{M}}$ and $\mathbf{M}$ be the predicted and ground truth entity linking affinity matrices, respectively. Then, we can simply write the entity linking loss as

$$\mathcal{L}_\text{EL}(\mathbf{L}, \hat{\mathbf{L}}, \mathbf{M}, \hat{\mathbf{M}}) = \text{BCE}(\mathbf{L}, \hat{\mathbf{L}}) + \beta \text{BCE}(\mathbf{M}, \hat{\mathbf{M}}), \tag{3}$$

where BCE denotes the binary cross-entropy loss.

Pre-training. We pre-train DocTr on a large-scale dataset of unlabeled document images. For simplicity of modeling, we only include one pre-training task, termed *masked detection modeling (MDM)*, for DocTr which we find sufficient for downstream tasks. Since pre-training is not the main focus of this work, we leave the exploration of other pre-training strategies [39, 11, 1] for future work. Fig. 4 illustrates MDM and compares it with related pre-training tasks. MDM is an extension of *masked vision-language modeling (MVLM)* [38, 39]. Both MDM and MVLM take OCR text and boxes as input. However, MVLM only randomly masks the text inputs. Instead, MDM randomly masks both the text inputs and their boxes. Specifically, we replace text with `[MASK]` and set boxes to `[0, 0, 0, 0]`. Then, we train DocTr to predict both the masked texts and their corresponding boxes. Note that this task is similar to object detection. Thus, the objective function can be written in the same way as Eq. (2), where the first term is for masked text classification, and the second term is for masked box regression. Also note that for MDM, the input image is not masked so that a model can better learn how to leverage the visual information to locate and identify the masked inputs.

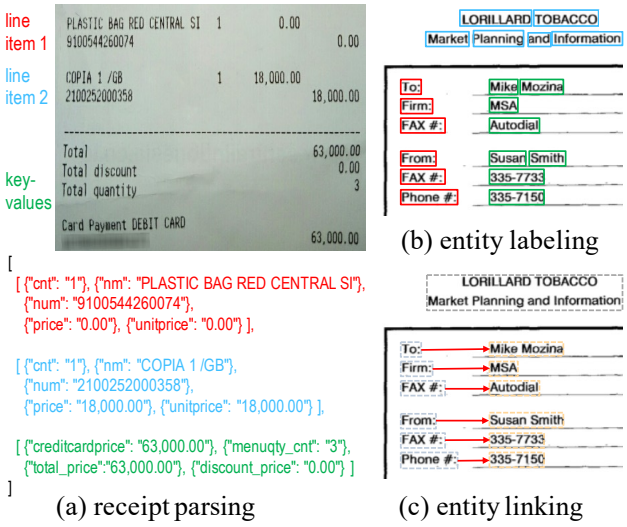


Figure 5: Illustration of the three tasks in the experiments.

model	F1
Donut [19]†	87.8
SPADE [15]	92.5
LayoutLMv2 [39] w/ IOB	91.4
BROS [11] w/ IOB	91.8
LayoutLMv3 [13] w/ IOB	92.2
LayoutLMv2 [39] w/ ours	92.7
BROS [11] w/ ours	92.9
LayoutLMv3 [13] w/ ours	93.6
DocTr (ours)	**94.4**

Table 1: Comparison with existing solutions on receipts parsing with the CORD dataset.

model	F1
SPADE [15]	41.7
BROS [11]	71.5
StructText [23]	44.1
DocTr (ours)	**73.9**

Table 2: Comparison with existing solutions on entity linking with the FUNSD dataset.

† We take the official model from [19] and report numbers using the metric from [15].

4. Experiments

Datasets and Tasks. We use three datasets in our experiments, IIT-CDIP document collection [21], CORD [27] and FUNSD [16]. We follow the convention in the literature [38, 39, 11, 1] to pre-train DocTr on the IIT-CDIP document collection, which is a large-scale dataset with 11 million unlabeled documents. CORD [27] is a receipt dataset with 800 training, 100 validation, and 100 testing samples. Each receipt in this dataset is labeled with a list of line items and key-value pair groups. FUNSD [16] consists of scanned forms, with 149 training and 50 testing examples. Each form is labeled with key/value entities together with links to indicate which keys and values are associated.

We evaluate our model's performance on three tasks, receipts parsing, entity labeling and entity linking. For *receipt parsing*, a model not only has to extract each receipt's entities but also correctly link entities to form line items and key-value pair groups. Fig. 5 (a) shows a sample receipt from CORD and its expected output after parsing. The sample contains two line items and four key-value pairs. For line items, it requires identifying each line item related entity (class and text) and group the entities of the same line item together. For key-value pairs, we identify class labels of the keys and return only text of the corresponding values. We use the same evaluation protocols and metrics as defined in [15] to evaluate the receipt parsing performance.

Entity labeling and *entity linking* are commonly adopted tasks [38, 15] to evaluate a pre-trained model's performance, which however are simplified versions of what we have defined in Sec. 3.1. Entity labeling requires assigning a class label to each word of the document. Fig. 5 (b) shows a sample from FUNSD where the task is to identify if a word belongs to a key (red), a value (green) or a title (blue). In entity linking, the assumption is that the key/value entities are correctly detected, and the task is to identify which keys and values should be linked (See Fig. 5 (c), red arrows). We evaluate entity labeling/linking by checking if the words/links are correctly labeled using F1-score as the metric.

4.1. Comparison with Existing Solutions

We compare DocTr with the existing methods on receipts parsing, entity labeling, and entity linking tasks, respectively.

For *receipts parsing*, SPADE [15] and Donut [19] are the only two other publicly available solutions (to the best of our knowledge) that address this task on CORD. The other existing general-purpose models [39, 11, 13] are not able to directly address this structured information extraction task out-of-the-box. For a fair comparison with our method, we fine-tune the officially released general-purpose models under two settings: using the standard IOB tagging for receipts parsing or using our proposed formulation. From Table 1, we can see that DocTr outperforms general-purpose models BROS, LayoutLMv2 and LayoutLMv3 by a noticeable margin when they are fine-tuned with the IOB tagging setting. When fine-tuned with our proposed formulation, the general-purpose models' performance improved but they are still behind DocTr, which shows the effectiveness of the proposed encoder-decoder solution for the anchor word based structure information extraction.

For *entity labeling*, we note that the majority of existing works (including DocTr) report numbers using word-level boxes (for position embedding) as input, and some others [22, 35, 13] use segment-level boxes from GT as input. Segment-level boxes provide more semantic information, and thus *their usage is unfair to those using word-level boxes.*. Since segment-level boxes are not conventional inputs (due to OCR limitations), in this experiments we mainly focus on comparing the methods with word-level boxes.

To address entity labeling, DocTr follows the general-

model	text box	FUNSD	CORD	#params
SPADE [15]	word	71.6	-	-
LayoutLM$_{\text{BASE}}$ [38].	word	78.7	94.7	113M
BROS$_{\text{BASE}}$ [11]	word	83.1	96.5	110M
DocFormer$_{\text{BASE}}$ [11]	word	83.3	96.3	183M
LayoutLMv2$_{\text{BASE}}$ [39]	word	82.8	95.0	200M
StructText [23]	word	83.4	-	107M
DocTr($ours$)	word	**84.0**	**98.2**	153M
LayoutLM$_{\text{LARGE}}$ [38]	word	79.0	95.0	343M
BROS$_{\text{LARGE}}$ [11]	word	84.5	**97.3**	340M
DocFormer$_{\text{LARGE}}$ [11]	word	84.5	97.0	536M
LayoutLMv2$_{\text{LARGE}}$ [39]	word	84.2	96.0	426M
FormNet [20]	word	**84.7**	**97.3**	345M
LiLT$_{\text{BASE}}$ [35]	segment	88.4	96.1	-
LayoutLMv3$_{\text{BASE}}$ [13]	segment	**90.3**	**96.6**	133M
StructualLM$_{\text{LARGE}}$ [22]	segment	85.1	-	426M
LayoutLMv3$_{\text{LARGE}}$ [13]	segment	**92.1**	**97.5**	368M

Table 3: Comparison with existing solutions on entity labeling (with FUNSD and CORD datasets).

formulation	text serial.	parsing (C)
IOB tagging [29]	raster scan	93.2
SPADE [15]	raster scan	93.0
DocTr (ours)	raster scan	**94.4**
IOB tagging [29]	oracle	94.1
SPADE [15]	oracle	93.9
DocTr (ours)	oracle	**95.0**

Table 4: Comparison of different SIE formulations under two text serialization settings, raster scan and oracle.

purpose models [38] to only fine-tune DocTr for IOB tagging and evaluate based on its IOB tagging outputs. This is less favorable for DocTr since the architecture and is dedicated to address our new formulation, and the pre-training strategy is not a main focus of this paper. However, we observe DocTr noticeably outperforming the existing solutions with comparable model sizes and text box embeddings ("Base" models with "word" text box embeddings in Table 3). Even when compared with larger pre-trained models, DocTr's performance is comparable or better on the CORD dataset.

For *entity linking*, we apply the objective introduced in Eq. (3) to train our model to link keys and values in FUNSD documents. We remove the entity extraction loss (Eq. (2)) but use ground truth entities as per the task definition. The results are shown in Table 2 – DocTr also outperforms the existing solutions by a noticeable margin in this task.

4.2. Model Properties

We analyze DocTr's design and consider other choices.

Problem Formulation. We use DocTr as the backbone network for the encoding of document inputs (image and

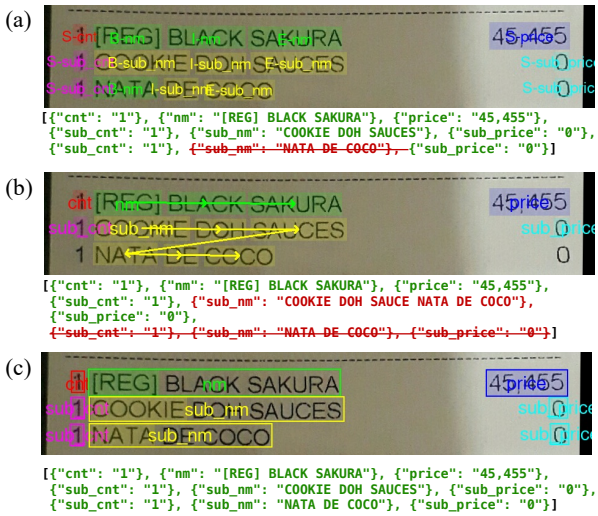


Figure 6: Visualization of receipt parsing results using different SIE formulations. Each result consists of the visualization of model predictions, and the parsing outputs (given the model predictions). **(a) IOB tagging** visualizes the predicted tags of OCR words. **(b) SPADE decoding** visualizes the decoded graph, and arrows between words indicate that the words are linked in the same entity. **(c) DocTr** visualizes the predicted anchor words and their bounding boxes. For the parsing outputs, green/red text means the predicted text matches/does not match ground truth. Strikethrough text means the ground truth text is missed from prediction.

OCR words) and apply different formulations to decode structured information. Specifically, we compare our formulation with IOB tagging and graph based solutions. For IOB tagging, we follow the literature [20, 38] and assign BIOES tags to each token and decode entities according to the tagged entity spans. Note that IOB tagging does not support entity linking. For a fair comparison, we link entities using a way similar to the anchor word association method introduced in Sec. 3.2. We treat the "B" tag or "S" tag of entities as the anchor words and link entities via decoding of entity linking affinity matrices. For graph based SIE, we follow the literature [11, 15] by attaching a SPADE [15] decoder at the end of DocTr. We fine-tune DocTr and decode graphs using the same way as specified in the original SPADE method. To understand the sensitivity of the SIE formulations with regard to the reading orders of input text, we evaluate them under two text serialization settings, raster scan and oracle. For oracle, we first order the ground truth entities in a raster scan manner, then order text while preserving the entity order.

Table 4 shows the receipt parsing results on the CORD dataset. Our proposed formulation achieves the best performance in both text serialization settings. We notice that, compared with the other two formulations, our formulation is less sensitive to text serialization with only 0.6 score drop (vs. 0.9 drop by IOB tagging or SPADE) while switching

anchor word	primary anchor	parsing (C)
first	name	94.2
last	name	94.1
first + last	first	94.0
first + last	name	**94.4**

Table 5: Receipt parsing (CORD) results under different choices of anchor words and primary anchors.

from oracle to raster scan text serialization. We also observe that our formulation can better address cases where there is dense text with multiple entities near each other. Fig. 6 shows an example visualization (see supplementary material for more results). For IOB tagging, it can tag most of the words well. However, even a single tagging error can cause failures of entity decoding, and an entity is missed from the parsing outputs. For SPADE, the dense words result in a challenge for constructing an entity graph, and the model incorrectly merges the two `sub_nm`'s as a single entity. In comparison, DocTr only requires identifying the anchor words which is an easier task and, with bounding box predictions, all the entities are correctly extracted.

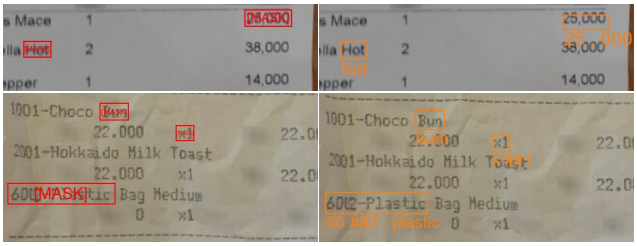
Anchor Word and Primary Anchor. We investigate different ways of designating anchor word and primary anchor. In Sec. 3.1, we introduced using the first word (in terms of reading order) of an entity as the anchor word. Here, we consider two alternatives: 1) using the last word or 2) both the first and last word as the anchor. Table 5 (row 1-2, 4) shows the comparison of these three choices. We notice that there is no significant differences (94.2 vs. 94.1) between using the first word and last word as the anchor word. Using both first and last as the anchor word gives slightly better performance. We hypothesize that this is because first and last words help better identify the boundary of an entity.

For primary anchor, we investigate its choices for line-item extraction. We consider two candidates: 1) using the anchor word of the first entity in a line-item, or 2) using the anchor word of `name` as the primary anchor. From Table 5 (row 3 and 4), we see the latter is a better choice with 0.4 improvement. This is reasonable since the first entity in a line-item may vary semantically (i.e., it could be `name`, `cnt` or other entity types), and thus it is harder to identify. However, this choice is also more flexible than using `name` as the primary anchor because there may be no `name` in an line-item. For CORD, each line-item always has a `name`, so this is not a concern (see supplementary material for primary anchor choices of other entity categories).

Pre-training. We evaluate the effectiveness of the pre-training task (MDM) introduced in Sec. 3.4. We consider three settings: 1) without pre-training, 2) with MVLM and 3) with MDM. Table 6 compares their performances. Without pre-training, the performance drops significantly. With MVLM, the performance improves but still falls behind using MDM. This shows the effectiveness of having MDM for document understanding pre-training. In particular, we see more benefit of using MDM for receipt parsing. This is because our proposed formulation requires bounding box regression, and *MDM helps learn better box predictions*.

pre-training	parsing (C)	ELB (F)	ELK (F)
none	82.3	14.2	12.0
MVLM [38]	90.9	82.7	73.0
MDM	**94.4**	**84.0**	**73.9**

Table 6: Receipt parsing (CORD), entity labeling (FUNSD) and entity linking (FUNSD) results using different pre-training tasks.



(a) inputs (b) predictions

Figure 7: Example pre-training predictions on CORD images. For inputs, we visualize masked word boxes, and their text is replace by `[MASK]`. For predictions, we visualize the predicted word boxes of the masked inputs. Under each box prediction, we also visualize its corresponding word token predictions.

	vis. enc.	VL-dec.	LCQ	parsing (C)	ELB (F)	ELK (F)
1)			N/A	92.3	82.1	73.2
2)		✓	✓	92.6	83.0	73.3
3)	✓	✓		90.7	14.9	9.5
4)	✓	✓	✓	**94.4**	**84.0**	**73.9**

Table 7: Impact of different architectural components: vision encoder, VL-decoder and language conditioned queries. ✓means the component is included in the architecture.

We also show example MDM pre-training predictions in Figure 7. Note that since both the input OCR box and text are masked, the model will need to not only predict what is masked but also predict where to find the masked word. We can see in most of the cases, the model can predict both kinds of information well. There are cases where the box (e.g., "25,000" in row 1) or text (e.g., "Bun" in row 2) is not accurately predicted. But the errors are reasonable. We also notice that the model can predict words that cannot be inferred through only text context, such as prices. This shows *the usage of visual information*.

Architecture Design. We then ablate the impact of the architectural components of DocTr. Table 7 shows the ablation results. The first row is a DocTr model with only the language encoder which is equivalent to the LayoutLM [38] model without visual inputs. The second row is a model with both the language encoder and VL-decoder but no vision encoder. These two models are close in performance. This is reasonable as without visual inputs the VL-decoder does not add much of information for decoding. Row 4 is the full model with both the vision encoder and VL-decoder. Compared with row 1 and 2, the performance improves noticeably. This suggests the importance of using visual information.

For row 3 and 4, we study the effectiveness of using the proposed language conditioned queries (LCQ). Specifically, we apply Eq. (1) to the cross-attention module when LCQ is checked. Otherwise, the standard cross-attention is used. We can see that LCQ is important since it helps to guide this one-to-one mapping between OCR and outputs, which is required by our proposed formulation.

5. Conclusion

We have presented a new approach for SIE from visually-rich documents. This approach is based on our novel formulation which includes object detection as part of the problem setting. This naturally leads us to include a transformer-based object detector as part of the architecture design and an object detection based loss in pre-training.

We have empirically shown that our proposed object detection based formulation readily addresses the structured information extraction task, and our solution outperforms existing solutions on SIE benchmarks. We hope this approach will initiate more efforts in combining object detection with existing vision-language models for document intelligence.

We also note that using anchor words limits the application of this approach to text-rich documents, and text-based entity extraction only. For future work, we explore solutions that extend the propose formulation for the extraction of non-textual content (e.g., symbols, logos, etc.) from documents.

References

[1] Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R Manmatha. Docformer: End-to-end transformer for document understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 993–1003, 2021. 2, 3, 4, 5, 6

[2] Ali Furkan Biten, Ron Litman, Yusheng Xie, Srikar Appalaraju, and R Manmatha. Latr: Layout-aware transformer for scene-text vqa. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16548–16558, 2022. 3

[3] Claire Cardie. Empirical methods in information extraction. *AI magazine*, 18(4):65–65, 1997. 1

[4] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pages 213–229. Springer, 2020. 4, 5

[5] Lei Cui, Yiheng Xu, Tengchao Lv, and Furu Wei. Document ai: Benchmarks, models and applications. *arXiv preprint arXiv:2111.08609*, 2021. 1

[6] Timo I Denk and Christian Reisswig. Bertgrid: Contextualized embedding for 2d document representation and understanding. *arXiv preprint arXiv:1909.04948*, 2019. 2

[7] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. In Jill Burstein, Christy Doran, and Thamar Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 4171–4186. Association for Computational Linguistics, 2019. 4, 5

[8] Dayne Freitag. Machine learning for information extraction in informal domains. *Machine learning*, 39(2):169–202, 2000. 1

[9] Chenyu Gao, Qi Zhu, Peng Wang, Hui Li, Yuliang Liu, Anton Van den Hengel, and Qi Wu. Structured multimodal attentions for textvqa. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 44(12):9603–9614, 2021. 3

[10] Łukasz Garncarek, Rafał Powalski, Tomasz Stanisławek, Bartosz Topolski, Piotr Halama, Michał Turski, and Filip Graliński. Lambert: Layout-aware language modeling for information extraction. In *International Conference on Document Analysis and Recognition*, pages 532–547. Springer, 2021. 2

[11] Teakgyu Hong, Donghyun Kim, Mingi Ji, Wonseok Hwang, Daehyun Nam, and Sungrae Park. Bros: A pre-trained language model focusing on text and layout for better key information extraction from documents. In *Proceedings of the AAAI Conference on Artificial Intelligence*, pages 10767–10775, 2022. 2, 5, 6, 7

[12] Ronghang Hu, Amanpreet Singh, Trevor Darrell, and Marcus Rohrbach. Iterative answer prediction with pointer-augmented multimodal transformers for textvqa. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9992–10002, 2020. 3

[13] Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. Layoutlmv3: Pre-training for document ai with unified text and image masking. *arXiv preprint arXiv:2204.08387*, 2022. 2, 6, 7

[14] Wonseok Hwang, Seonghyeon Kim, Minjoon Seo, Jinyeong Yim, Seunghyun Park, Sungrae Park, Junyeop Lee, Bado Lee, and Hwalsuk Lee. Post-ocr parsing: building simple and robust parser via bio tagging. In *Workshop on Document Intelligence at NeurIPS 2019*, 2019. 2

[15] Wonseok Hwang, Jinyeong Yim, Seunghyun Park, Sohee Yang, and Minjoon Seo. Spatial dependency parsing for semi-structured document information extraction. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 330–343, 2021. 1, 2, 6, 7

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

[17] Justin Johnson, Ranjay Krishna, Michael Stark, Li-Jia Li, David Shamma, Michael Bernstein, and Li Fei-Fei. Image retrieval using scene graphs. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3668–3678, 2015. 3

[18] Anoop Raveendra Katti, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul. Chargrid: Towards understanding 2d documents. *arXiv preprint arXiv:1809.08799*, 2018. 2

[19] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pages 498–517. Springer, 2022. 2, 6

[20] Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. Formnet: Structural encoding beyond sequential modeling in form document information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3735–3754, 2022. 2, 7

[21] David Lewis, Gady Agam, Shlomo Argamon, Ophir Frieder, David Grossman, and Jefferson Heard. Building a test collection for complex document information processing. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 665–666, 2006. 6

[22] Chenliang Li, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. 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 (Volume 1: Long Papers)*, pages 6309–6318, 2021. 6, 7

[23] Yulin Li, Yuxi Qian, Yuechen Yu, Xiameng Qin, Chengquan Zhang, Yan Liu, Kun Yao, Junyu Han, Jingtuo Liu, and Errui Ding. Structext: Structured text understanding with multi-modal transformers. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 1912–1920, 2021. 2, 3, 4, 6, 7

[24] Xiaojing Liu, Feiyu Gao, Qiong Zhang, and Huasha Zhao. Graph convolution for multimodal information extraction from visually rich documents. *arXiv preprint arXiv:1903.11279*, 2019. 2

[25] 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. *arXiv preprint arXiv:1907.11692*, 2019. 2

[26] Ahmed Nassar, Nikolaos Livathinos, Maksym Lysak, and Peter Staar. Tableformer: Table structure understanding with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4614–4623, 2022. 2, 3

[27] Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. Cord: a consolidated receipt dataset for post-ocr parsing. In *Workshop on Document Intelligence at NeurIPS 2019*, 2019. 6

[28] Devashish Prasad, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita Sultanpure. Cascadetabnet: An approach for end to end table detection and structure recognition from image-based documents. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, pages 572–573, 2020. 2, 3

[29] Lance A Ramshaw and Mitchell P Marcus. Text chunking using transformation-based learning. In *Natural language processing using very large corpora*, pages 157–176. Springer, 1999. 1, 2, 7

[30] Ellen Riloff. Automatically generating extraction patterns from untagged text. In *Proceedings of the national conference on artificial intelligence*, pages 1044–1049, 1996. 1

[31] Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. Deepdesrt: Deep learning for detection and structure recognition of tables in document images. In *2017 14th IAPR international conference on document analysis and recognition (ICDAR)*, volume 1, pages 1162–1167. IEEE, 2017. 2

[32] Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. Towards vqa models that can read. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 8317–8326, 2019. 3

[33] Kaihua Tang, Yulei Niu, Jianqiang Huang, Jiaxin Shi, and Hanwang Zhang. Unbiased scene graph generation from biased training. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 3716–3725, 2020. 3

[34] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need. *Advances in neural information processing systems*, 30, 2017. 2, 4, 5

[35] Jiapeng Wang, Lianwen Jin, and Kai Ding. Lilt: A simple yet effective language-independent layout transformer for structured document understanding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 7747–7757, 2022. 6, 7

[36] Mengxi Wei, Yifan He, and Qiong Zhang. Robust layout-aware ie for visually rich documents with pre-trained language models. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2367–2376, 2020. 2

[37] Danfei Xu, Yuke Zhu, Christopher B Choy, and Li Fei-Fei. Scene graph generation by iterative message passing. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5410–5419, 2017. 3

[38] Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. Layoutlm: Pre-training of text and layout

for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1192–1200, 2020. 2, 4, 5, 6, 7, 8, 9

[39] Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, et al. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2579–2591, 2021. 2, 3, 4, 5, 6, 7

[40] Jianwei Yang, Jiasen Lu, Stefan Lee, Dhruv Batra, and Devi Parikh. Graph r-cnn for scene graph generation. In *Proceedings of the European conference on computer vision (ECCV)*, pages 670–685, 2018. 3

[41] 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 *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5315–5324, 2017. 2

[42] Wenwen Yu, Ning Lu, Xianbiao Qi, Ping Gong, and Rong Xiao. Pick: processing key information extraction from documents using improved graph learning-convolutional networks. In *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4363–4370. IEEE, 2021. 2

[43] Rowan Zellers, Mark Yatskar, Sam Thomson, and Yejin Choi. Neural motifs: Scene graph parsing with global context. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5831–5840, 2018. 3

[44] Peng Zhang, Yunlu Xu, Zhanzhan Cheng, Shiliang Pu, Jing Lu, Liang Qiao, Yi Niu, and Fei Wu. Trie: end-to-end text reading and information extraction for document understanding. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 1413–1422, 2020. 2

[45] Xinyi Zheng, Douglas Burdick, Lucian Popa, Xu Zhong, and Nancy Xin Ru Wang. Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 697–706, 2021. 2, 3

[46] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable detr: Deformable transformers for end-to-end object detection. In *International Conference on Learning Representations*, 2020. 4, 5