See then Tell: Enhancing Key Information Extraction with Vision Grounding

Anonymous ACL submission

Abstract

In the digital era, understanding visually rich documents that combine text, complex layouts, and imagery is crucial. Traditional Key Information Extraction (KIE) methods rely on Optical Character Recognition (OCR), which often incurs latency, computational overhead, and errors. Recent image-to-text approaches bypass OCR but typically yield plain text outputs without vision grounding. In this paper, we introduce STNet (**S**ee then **T**ell Net), an end-to-end model that jointly produces accurate textual answers and their corresponding vision grounding. At the core of STNet lies a novel <see> token, prepended to each response. During generation, <see> directs the model first to *see* — observing the regions of the image related to the input question (decoded into physical coordinates) — and then to *tell*, emitting the textual answer. To enhance the model's *see* capabilities, we collect extensive structured table recognition datasets and leverage GPT-4 to develop TVG (**T**ableQA with **V**ision **G**rounding), a dataset of QA pairs annotated with vision grounding. Our approach demonstrates substantial advancements in KIE performance, achieving state-of-the-art results on publicly available datasets such as CORD, SROIE, and DocVQA. The code and dataset will be made publicly available.

1 Introduction

Visually rich document is a type of medium that centers around text while also incorporating the layout and related visual imagery. In the digital information age, many documents are digitized and saved as images. Understanding document images plays a pivotal role across multiple domains such as document analysis (Cui et al., 2021), document retrieval (Mass and Roitman, 2020), and office robotic process automation (Axmann and Harmoko, 2020). This comprehension significantly bolsters the efficiency and accuracy of information processing. Key Information Extraction (KIE) aims to locate, analyze, and extract key entities (like names, dates, and numerical data) from documents. KIE has become a key part of the document understanding field.

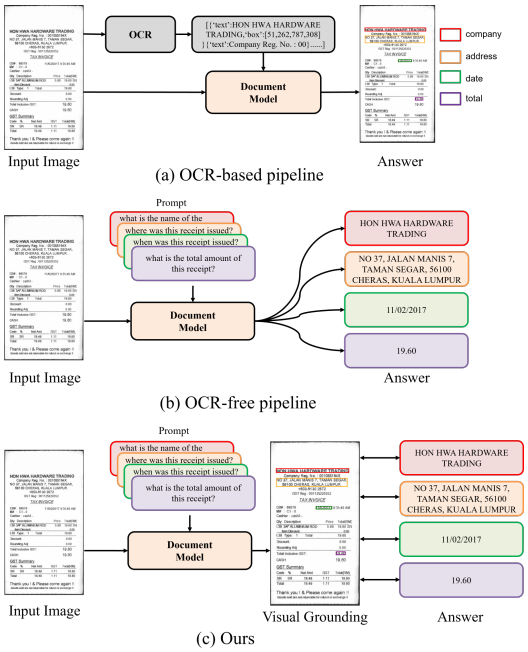


Figure 1: The illustration of contemporary KIE pipelines.

As illustrated in Figure 1, existing methods for KIE can be broadly categorized into two groups. Traditional methods (Xu et al., 2020, 2021; Huang et al., 2022; Zhang et al., 2023b; Appalaraju et al., 2021) rely on Optical Character Recognition (OCR) engines to first extract text and coordinate information from document images, which are then fed into a document model for analysis and classification. However, this pipeline is heavily dependent on the OCR engine, resulting in additional latency and computational costs. Furthermore, errors originating from the OCR step can propagate to the document model, thereby deteriorating overall performance. Recent advancements in document un-

derstanding (Kim et al., 2022; Cao et al., 2023; Okamoto et al., 2024) have introduced end-to-end image-to-text paradigms. These methods enable document models to directly process document images without explicit OCR. To achieve this, they leverage the Transformer architecture (Vaswani et al., 2017) to decode OCR results during the pre-training stage, endowing the document model with reading capabilities. These OCR-free approaches show strong performance across various document understanding tasks. Nevertheless, the KIE task differs from typical Visual Question Answering (VQA) tasks (Singh et al., 2019; Lu et al., 2022) due to the strong correspondence required between the extracted information and the visual content of the document. Therefore, it is essential to design a module that aligns the predicted plain text answers with the visual content.

Currently, most document KIE datasets, such as DocVQA (Mathew et al., 2021) and WikiTableQuestions (Pasupat and Liang, 2015), purely offer simple plain text Question Answering (QA) pairs without vision grounding for each answer within the image context. With the rise of large language models (LLMs) like ChatGPT (Brown et al., 2020) and GPT-4 (OpenAI, 2023), recent work (Han et al., 2023; Zhang et al., 2023a) has explored using LLMs to generate domain-specific instruction tuning data. Inspired by this, we present an automated processing pipeline that leverages GPT-4 to generate KIE data with robust vision grounding for the document domain. This is expected to enhance KIE performance by providing precise and dependable vision grounding.

In this work, we introduce a novel end-to-end model named STNet (**S**ee then **T**ell Net), which can simultaneously provide answers and corresponding vision grounding. Unlike existing methods, we explicitly design a <see> token to guide the model in identifying the relevant location within the image. Accordingly, we develop a specialized physical decoder to interpret the physical coordinate associated with the <see> token. In downstream tasks, we simply place the <see> token at the beginning of the answer text, thereby providing vision grounding for the answer. To further enhance the model's *see* capabilities, we collect a substantial number of highly structured table recognition datasets, such as PubTables1M (Smock et al., 2022) and iFLYTAB (Zhang et al., 2024). Leveraging the powerful text understanding capabilities of GPT-4, we construct a TVG (**T**ableQA with **V**ision **G**rounding) dataset. This dataset not only provides the related plain text QA pairs but also includes the specific vision grounding of the QA pairs within the image. We validate STNet on publicly available datasets such as CORD (Park et al., 2019), SROIE (Huang et al., 2021), and DocVQA (Mathew et al., 2021), achieving state-of-the-art results. The main contributions of this paper are as follows:

- We introduce STNet, a novel end-to-end model that not only provides textual answers but also excels in offering vision grounding, enabled by our specially designed <see> token and specialized physical decoder.
- We introduce a GPT-4-driven automated QA pair generation method, creating the TVG dataset. This dataset comprises QA pairs with precise vision grounding, essential for enhancing visual comprehension.
- Experimental results on publicly available datasets such as CORD, SROIE, and DocVQA demonstrate that our STNet model achieves state-of-the-art performance.

2 Related Work

Early KIE algorithms used rule-based methods, heavily relying on prior knowledge. These approaches were limited to fixed-format documents and lacked robustness for diverse real-world applications. The rapid development of deep learning has brought superior solutions to document understanding, which can be primarily categorized into OCR-based and OCR-free methods.

2.1 OCR-based Methods

OCR-based methods require an OCR engine to extract text and coordinate information from visually rich document images as input for document models. The LayoutLM family (Xu et al., 2020) introduces a pre-training framework that combines text and layout features, with LayoutLMv2 (Xu et al., 2021) enhancing representation capabilities through spatial-aware self-attention and tasks like text-image alignment and text-image matching. LayoutLMv3 (Huang et al., 2022) further advances this approach by reducing visual feature extraction costs with patch encoding and introducing masked image modeling and word-patch alignment tasks. DocFormer (Appalaraju et al., 2021) integrates visual and spatial information into each Transformer layer using a self-attention encoder.

GraphDoc (Zhang et al., 2023b) employs BERT and Swin Transformer for semantic and visual encoding, respectively, with an attention-based graph network for localized feature interaction. Despite their commendable performance, these models exhibit a significant reliance on OCR tools, making them vulnerable to cascading OCR errors that can severely degrade overall performance.

2.2 OCR-free Methods

OCR-free methods aim to remove reliance on OCR modules, enabling faster inference with fewer parameters. For example, Donut (Kim et al., 2022) uses the Swin Transformer to encode image patches and BART-like Transformers to generate text sequences, introducing a prompt mechanism to switch between tasks. This end-to-end approach simplifies the model architecture and achieves cost-effectiveness by directly mapping input images into structured outputs. Pix2Struct (Lee et al., 2023) extends these improvements by scaling up pre-training data and tasks, while SeRum (Cao et al., 2023) employs selective region concentration to enhance precision and speed. These methods streamline information processing and accelerate reasoning, making them highly effective in KIE. Recently, some researchers have also attempted to provide vision grounding for answers, further improving the effectiveness of these models. UNITS (Kil et al., 2023) outputs text coordinates as a part of the sequence. However, this explicit approach results in overly long sequences and increased error accumulation. CREPE (Okamoto et al., 2024) employs a multi-head architecture, where a specialized head is designed to predict implicit text coordinates from the `</ocr>` token following the answer text. However, using generated text to assist in location inference ("tell then see") provides limited improvement for KIE itself, as will be validated through comparisons with our "see then tell" approach.

2.3 Leveraging LLMs for Datasets

High-quality datasets are crucial for improving model performance, yet existing ones often fail to meet emerging needs, and manual annotation is costly. With the advent of LLMs like ChatGPT (Brown et al., 2020) and GPT-4 (OpenAI, 2023), researchers have begun leveraging their robust linguistic and coding capabilities to efficiently process large volumes of data and construct new datasets. WizardLM (Xu et al., 2023) and GPT-4-LLM (Peng et al., 2023) have validated the effectiveness of using ChatGPT and GPT-4 to generate instruction fine-tuning datasets. ChartLlama (Han et al., 2023) utilizes GPT-4 to generate chart images along with diverse and precise QA pairs, thereby aiding in the training of models for comprehensive chart understanding. TableLlama (Zhang et al., 2023a) has achieved similar advancements in the domain of table data. However, the WizardLM and GPT-4-LLM datasets are restricted to textual data only. While TableLlama and ChartLlama are multimodal, they lack spatial annotations for the QA pairs, limiting their utility for vision grounding. To address these limitations, we propose a novel automated QA pair generation method to construct the TVG dataset, which includes QA pairs along with their corresponding spatial information.

3 Task Definition

Given a document image $\boldsymbol{I}$ and a question sequence $\boldsymbol{Q} = \{\boldsymbol{q}_i \in \mathbb{R}^v \mid i = 1, \ldots, T_q\}$ as a prompt, our objective is to enable the model to predict the answer sequence $\boldsymbol{A} = \{\boldsymbol{a}_i \in \mathbb{R}^v \mid i = 1, \ldots, T_a\}$ for completing KIE. Here, T_q denotes the length of the question sequence, T_a denotes the length of the answer sequence, and v is the size of the token vocabulary. Previous methods have achieved remarkable results using this format. In contrast, we divide this process into two distinct phases: see and tell. In the see phase, we output a `<see>` token that implicitly encodes the physical coordinates $\boldsymbol{p} = \{p_j \in \mathbb{N} \mid j = 1, \ldots, 8\}$, which represent a four-point polygon, defining the physical location in the document image associated with $\boldsymbol{Q}$. This differs from logical locations, such as row and column coordinates. We choose four-point polygons over bounding boxes, as polygons more effectively handle warped or rotated text. Subsequently, in the *tell* phase, the model generates the answer text following the `<see>` token.

4 Methodology

As illustrated in Figure 2, STNet is built on Donut (Kim et al., 2022) and consists of two primary modules: a vision encoder and a text decoder. The vision encoder is tasked with processing image features which are subsequently interpreted by the text decoder to formulate the answer sequence $\boldsymbol{A}$. Our model introduces a novel `<see>` token that implicitly encodes physical coordinates at the beginning of $\boldsymbol{A}$. Specifically, we design a dedicated physical decoder to extract these coordinates and propose a

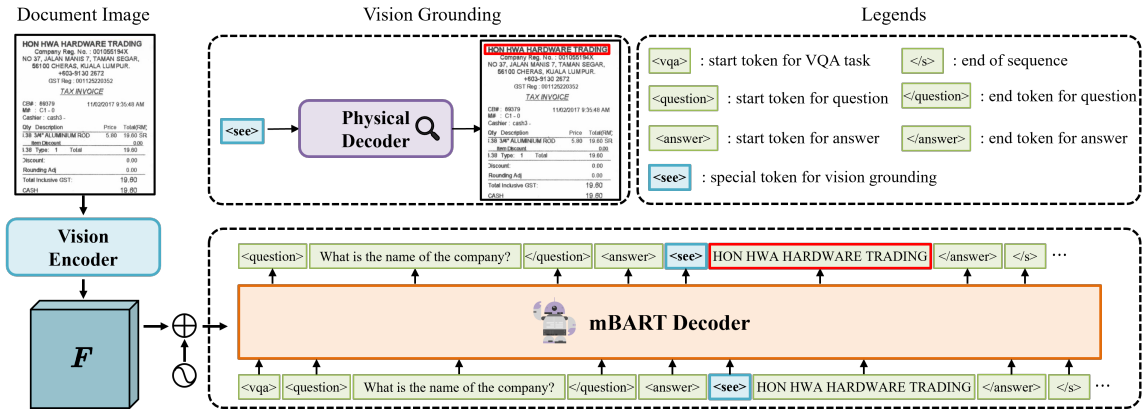


Figure 2: The overall architecture of STNet. It mainly consists of a vision encoder and a text decoder. Our text decoder's output answer sequence includes a special <see> token. The physical decoder is designed to decode the hidden states corresponding to the <see> token to obtain coordinates for vision grounding.

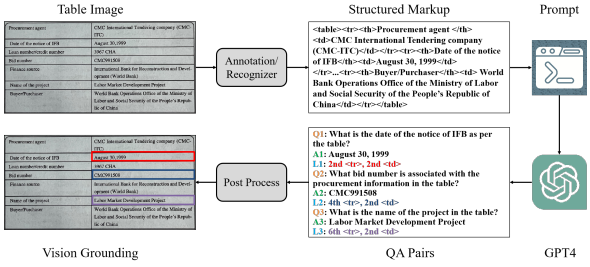


Figure 3: TVG dataset construction pipeline.

corresponding *see* loss to supervise this encoding. The <see> token with physical location information effectively guides the subsequent output of the answer text, ensuring "see then tell". Due to the lack of suitable datasets to train <see>, we accumulate a large amount of structured table recognition data and construct the TVG dataset utilizing GPT-4. More details are elaborated in subsequent sections.

4.1 Vision Encoder

The vision encoder transforms the input document image $\boldsymbol{I}$ into a feature map $\boldsymbol{F} \in \mathbb{R}^{H \times W \times D}$. This feature map is subsequently serialized into a set of embeddings $\boldsymbol{Z} = \{\boldsymbol{z}_i \in \mathbb{R}^D \mid i = 1, \ldots, N\}$, where N represents the size of the feature map and D is the dimension of the encoder's latent vectors. Following Donut, we adopt the Swin Transformer (Liu et al., 2021) as our primary vision backbone due to its superior performance demonstrated in previous studies. Additionally, we incorporate positional encoding (Vaswani et al., 2017) into $\boldsymbol{F}$ to produce the final vision embeddings $\boldsymbol{Z}$, enhancing the model's perception of location.

4.2 Text Decoder

Similar to Donut, we utilize the BART (Lewis et al., 2020) decoder to generate the answer sequence $\boldsymbol{A}$, which is conditioned on the $\boldsymbol{Z}$ and prompted by the question sequence $\boldsymbol{Q}$. Since STNet is trained to predict the next token like LLMs (OpenAI, 2023), the training objective is to minimize the negative log-likelihood of the target sequence.

$$\mathscr{L}_{\text{lm}} = -\frac{1}{T_a} \sum_{i=1}^{T_a} \log P\left(\boldsymbol{a}_i \mid \boldsymbol{Z}, \boldsymbol{Q}, \boldsymbol{a}_{1:i}\right) \quad (1)$$

4.3 Physical Decoder

We explore the fundamental human cognitive process of "see then tell", where individuals first *see* — gathering visual information and contextual insights — and then *tell* — constructing responses. This sequence notably enhances the accuracy and relevance of interactions. To effectively mirror this intuitive cognitive pattern, our proposed method adopts a two-phase output strategy. The <see> token initiates the *see* phase to perceive location information related to $\boldsymbol{Q}$ in the document image, followed by the *tell* phase that outputs the answer text. We design a physical decoder that decodes the hidden states $\boldsymbol{H} = \{\boldsymbol{h}_i \in \mathbb{R}^D \mid i = 1, \ldots, T\}$ extracted from the final layer of our text decoder, specifically corresponding to the <see> token, allowing us to obtain the polygon coordinates $\boldsymbol{p}$ within the image context for vision grounding. To facilitate this prediction, we employ a quantization strategy utilizing a specialized vocabulary composed of 1,000 unique tokens, ranging from <0> to <999>, collectively denoted as $\boldsymbol{Loc} \in \mathbb{R}^{1000 \times D}$.

For each coordinate p_j within a polygon $\boldsymbol{p}_i$, its associated hidden state $\boldsymbol{h}_i$ undergoes a linear transformation, producing $\boldsymbol{h}_i^{p_j}$ as a query against the vocabulary $\boldsymbol{Loc}$. The final determination of p_j's position is computed based on the expected location derived from the probability distribution over $\boldsymbol{Loc}$, as provided by $\boldsymbol{h}_i^{p_j}$, divergent from previous direct classification methods (Chen et al., 2022) over a location vocabulary:

$$\boldsymbol{h}_i^{p_j} = \text{Linear}\left(\boldsymbol{h}_i\right) \tag{2}$$

$$\boldsymbol{b}^{p_j} = \text{softmax}\left(\boldsymbol{h}_i^{p_j}\boldsymbol{Loc}^\top\right) \tag{3}$$

$$E\left(p_j\right) = \sum_{i=0}^{999} i \cdot b_i^{p_j} \tag{4}$$

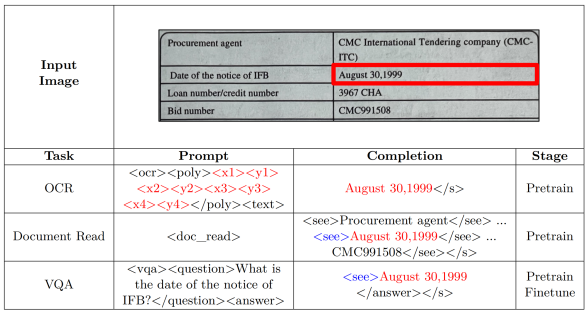
Here, $\boldsymbol{b}^{p_j} \in \mathbb{R}^{1000}$ represents the probability distribution for the position of p_j. The polygon regression loss of *see* is defined as follows:

$$\mathscr{L}_{\text{see}} = \frac{1}{8}\sum_{j=1}^{8}\left(E\left(p_j\right) - p_j^*\right)^2 \tag{5}$$

where p_j^* represents the ground truth label.

4.4 TVG Dataset

Current document datasets, such as DocVQA (Mathew et al., 2021) and WikiTableQuestion (Pasupat and Liang, 2015), only offer plain text QA pairs, constraining the enhancement of STNet's ability to *see*. Inspired by recent studies (Xu et al., 2023; Han et al., 2023; Zhang et al., 2023a) which leverage LLMs for dataset construction, we propose a comprehensive GPT-4-based method, as depicted in Figure 3, to automatically construct QA datasets for document images. Tables, as a special form of document image, can be described using structured markup such as HTML, and high-quality table data can be readily sourced from online resources. To this end, we have collected a number of table recognition datasets including PubTables1M (Smock et al., 2022) and iFLYTAB (Zhang et al., 2024), each sample containing both images and their corresponding structured annotation. We design prompt templates and feed structured markup into GPT-4 to generate QA pairs, each with a logical location (row, column) of the answer. This location is used to retrieve the corresponding cell from annotations, and only QA pairs with answers matching the ground truth are retained. Finally, we map logical coordinates to physical locations (cell polygon boxes) using the original annotations, linking each answer to its position in the image. Consequently, we construct the TVG dataset, which includes table images $\boldsymbol{I}$ and the QA pairs with physical location annotations $\{\boldsymbol{Q}, \boldsymbol{A}, \boldsymbol{p}\}$. More details can be found in our Appendix.



Task	Prompt	Completion	Stage
OCR	<ocr><poly><x1><y1><x2><y2><x3><y3><x4><y4></poly><text>	August 30,1999</s>	Pretrain
Document Read	<doc_read>	<see>Procurement agent</see> ... <see>August 30,1999</see> ... CMC991508</see></s>	Pretrain
VQA	<vqa><question>What is the date of the notice of IFB?</question><answer>	<see>August 30,1999</answer></s>	Pretrain Finetune

Figure 4: The illustration of the task design.

5 Implementation Details

5.1 Pre-training Tasks

To bolster STNet's capability to perceive text locations — essentially, its ability to *see* — we have integrated a multi-task pre-training strategy encompassing three distinct sub-tasks: OCR, Document Read, and VQA, as illustrated in Figure 4.

5.2 Training Strategy

In the initial pre-training phase, in addition to the previously constructed TVG dataset and its data sources — the training sets from PubTables1M (Smock et al., 2022) and iFLYTAB (Zhang et al., 2024) — we also employ a synthetic dataset comprising 2.2 million entries in both Chinese and English from SynthDog (Kim et al., 2022). PubTables1M, iFLYTAB, and SynthDog are used for OCR and document read training, while TVG is utilized for OCR and VQA tasks.

After pre-training, STNet is fine-tuned on specialized datasets for KIE. Each dataset is tailored to meet the VQA task specifications and is combined with TVG at a 1:1 ratio, with consistent supervision of the *see* loss on TVG. It ensures that the output <see> token retains its physical location perception ability to guide the output of answer text, even without *see* loss supervision on downstream datasets.

STNet utilizes two types of loss: $\mathscr{L}_{\text{lm}}$ and $\mathscr{L}_{\text{see}}$. The total loss is computed as a weighted sum of these components.

$$\mathscr{L}_{\text{total}} = \mathscr{L}_{\text{lm}} + \lambda\mathscr{L}_{\text{see}} \tag{6}$$

After extensive evaluation, we set $\lambda = 0.001$.

5.3 Inference

During the inference phase, we feed the question sequence $\boldsymbol{Q}$ to STNet as a prompt, guiding it to output the answer sequence $\boldsymbol{A}$. Utilizing the hidden states $\boldsymbol{H}$ from the text decoder's last layer, we can decode the polygon information $\boldsymbol{p}$, associated with the <see> token preceding each response in $\boldsymbol{A}$. Through the visualization of this vision grounding, we can enhance the interpretability of the answers.

6 Experiments

6.1 Evaluation Benchmarks and Metrics

To fully demonstrate the effectiveness of STNet, we conduct experiments on three benchmark datasets. The SROIE (Huang et al., 2021) dataset contains 973 scanned receipts, annotated with four fields. Performance is measured using field-level F1 (Hwang et al., 2019) and TED accuracy (Zhong et al., 2020). The CORD (Park et al., 2019) dataset includes 1,000 receipts with 30 entity types, and uses the same evaluation metrics as SROIE. The DocVQA (Mathew et al., 2021) dataset comprises 50,000 QA pairs from over 12,000 document pages. Due to missing ground truth in the test set, evaluations are performed on the validation set using ANLS (Average Normalized Levenshtein Similarity). More details can be found in the Appendix.

6.2 Results

As shown in Table 1, we compare our approach with various prior methods. The STNet* model employs supervised training on the *see* process for downstream datasets by mapping answer texts to corresponding coordinates. Conversely, the original STNet omits this step and infers answer coordinates using the physical decoder pre-trained on TVG, ensuring a fair comparison. OCR-based methods are evaluated using text and bounding boxes extracted by OCR engines, following the approach in Donut (Kim et al., 2022) and SeRum (Cao et al., 2023), ensuring fair model comparisons. LayoutLMv3* (Huang et al., 2022) denotes results from its original paper, which uses ground-truth annotations for OCR during evaluation. More details can be found in the Appendix.

Our approach achieves state-of-the-art performance on both SROIE and CORD datasets. While LayoutLMv3* performs well using ground-truth annotations for OCR, its performance degrades significantly due to cascading OCR errors when using real OCR outputs, as shown in the LayoutLMv3 results. In contrast, STNet's end-to-end design eliminates reliance on OCR modules and costly high-precision annotations, making it more efficient for training and inference.

On the DocVQA dataset, where answer text location information is unavailable, we report results only for the original STNet. It achieves the second-best performance, trailing the OCR-based LayoutLMv2 (Xu et al., 2021). This dataset's complex and densely packed text makes model performance highly sensitive to resolution. Our STNet standardizes input image resolution to 1280×960 for consistent processing, whereas LayoutLMv2 leverages the advanced Microsoft Read API[1] to extract text and bounding boxes.

STNet not only achieves state-of-the-art results among OCR-free methods but also provides vision grounding for the answers, interacting better with the real world. This is particularly notable in datasets like DocVQA, which lack answer text location information. Figure 5 showcases the outcomes of text coordinates acquisition by our model.

6.3 Generalization to Advanced MLLMs

To assess the generalizability of our approach, we apply it to advanced multimodal large language models (MLLMs) such as Qwen2-VL (Wang et al., 2024). We integrate our vision grounding module into the MLLM architecture and augment the prompt with the <see> token. The model is trained on the TVG dataset with explicit *see* supervision on <see> to acquire *see* capabilities. During evaluation, we follow the zero-shot prompt setting adopted in prior MLLM-based KIE studies (Luo et al., 2024). Specifically, for key-value annotations in CORD, we prompt the model with questions in the form of "Q: What is the 'key'? <see> A: 'value'". To ensure fair comparison, we adopt the same filtering strategy as previous work, removing samples where a single entity corresponds to multiple values. Evaluation is conducted using ANLS. More details can be found in the Appendix.

As shown in Table 2, our method yields consistent performance gains on both Qwen2-VL-2B and Qwen2-VL-7B, surpassing the improvements achieved by previous methods such as RIDGE (Jiang et al., 2025). These results demonstrate the strong generalizability of our approach.

[1]https://docs.microsoft.com/en-us/azure/cognitive-

Method	OCR	CORD		SROIE		DocVQA
		F1	Acc.	F1	Acc.	ANLS
BROS (Hong et al., 2022)	✔	74.7	70.0	-	-	-
LayoutLMv2 (Xu et al., 2021)	✔	78.9	82.4	61.0	91.1	**74.2**
LayoutLMv3 (Huang et al., 2022)	✔	80.5	87.8	65.0	92.7	-
LayoutLMv3* (Huang et al., 2022)	✔	**96.6**	-	-	-	-
Donut (Kim et al., 2022)	✘	84.1	90.9	83.2	92.8	59.7
SeRum (Cao et al., 2023)	✘	84.9	91.5	85.8	95.4	-
CREPE (Okamoto et al., 2024)	✘	85.0	-	-	-	58.4
STNet	✘	88.1	92.3	87.8	97.1	63.7
STNet*	✘	88.8	**93.5**	**88.3**	**97.4**	-

Table 1: Comparison with SOTA methods across different datasets. The field-level F1 scores and tree-edit-distance-based accuracies are reported. **Bold** indicates the best result. Underline indicates the second best. LayoutLMv3* represents the results reported in the original paper, where ground-truth annotations for OCR are used during evaluation. More details can be found in the Appendix.

Model	CORD	SROIE	DocVQA
Qwen2-VL-7B	80.40	97.50	91.66
+ RIDGE	85.53	97.74	-
+ *see*	85.83	97.92	91.97
Qwen2-VL-2B	76.76	92.64	84.80
+ *see*	79.59	93.79	86.80

Table 2: Performance improvements in zero-shot KIE for MLLMs. All results are evaluated using ANLS.

λ	$1e^{-1}$	$1e^{-2}$	$1e^{-3}$	$1e^{-4}$
F1	86.4	88.0	88.3	88.0

Table 3: Comparison of STNet's performance under different *see* loss weights λ.

System	TVG	*See*	SS	F1	Acc.
T1	✘	✘	✘	84.7	94.1
T2	✔	✘	✘	86.2	96.3
T3	✔	✔	✘	87.8	97.1
T4	✔	✔	✔	88.3	97.4

Table 4: Results of the evaluation for the STNet model on the SROIE dataset. "TVG" denotes the use of the TVG dataset. *See* indicates the inclusion of the <see> token, and "SS" signifies the use of *see* supervision on SROIE. "F1" and "Acc." correspond to the performance metrics on the SROIE test set.

6.4 Ablation Study

To validate the effectiveness of each of our contributions, we build systems T1 through T4 based on STNet, which is built on the Donut architecture. We evaluate all systems on the SROIE dataset.

6.4.1 Impact of See Loss Weight.

During STNet training, the total loss is computed as a weighted combination of $\mathscr{L}_{\text{lm}}$ and $\mathscr{L}_{\text{see}}$, which differ significantly in scale. Determining the optimal weight λ for the *see* loss is crucial for balancing these losses. As shown in Table 3, $\lambda = 0.001$ achieves the best performance.

6.4.2 The Effectiveness of the TVG Dataset.

The TVG dataset is designed to enhance the training efficacy of our models. To determine whether performance improvements stem solely from extended pre-training, we conduct two experimental setups, T1 and T2, as detailed in Table 4. The results show that T2 significantly outperforms T1, validating the effectiveness of the TVG dataset.

6.4.3 The Effectiveness of "See then Tell".

As shown in Figure 5, our STNet is capable of providing vision grounding for the answers, highlighting its ability to *see*. To evaluate whether this ability results in improved extraction accuracy, we compare the performance of T2 and T3, as shown in Table 4. The results indicate a marked improvement in T3 over T2, which doesn't generate <see> embedded with text coordinates for physical location perception to guide the output of answer text. This validates the effectiveness of "see then tell".

6.4.4 Impact of See Supervision.

The calculation of *see* loss requires answer location information, which is unavailable in many document datasets, such as DocVQA. To assess the model's reliance on *see* supervision for downstream datasets, we conduct a comparative analysis between T3 and T4. As shown in Table 4, T3 per-

services/computer-vision/concept-recognizing-text

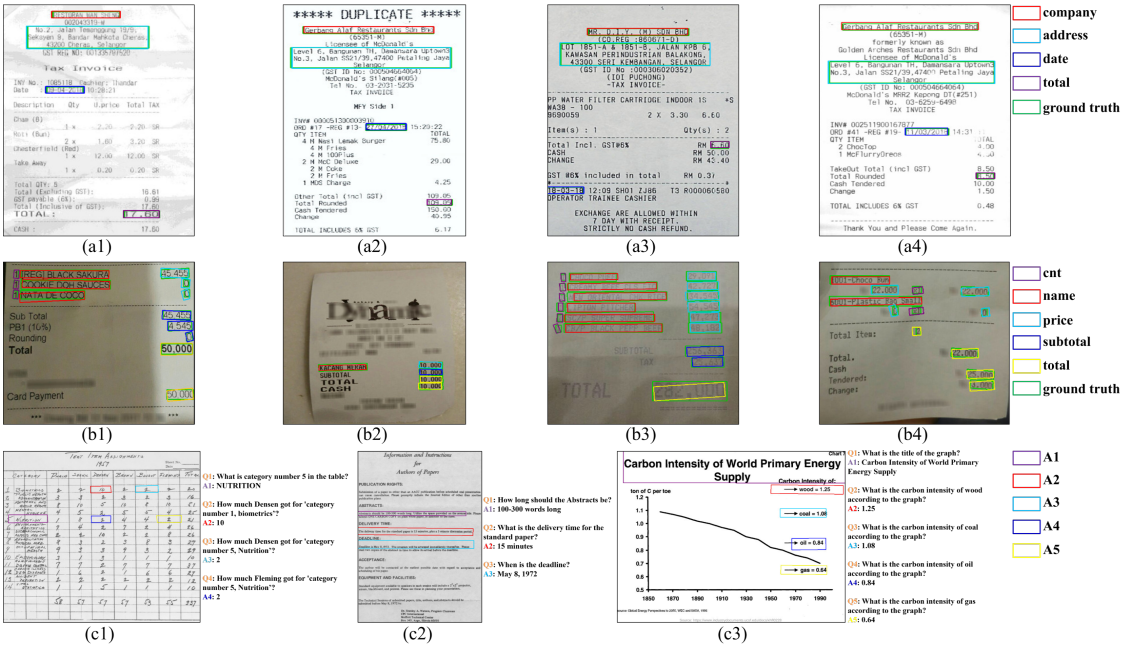


Figure 5: The results of acquiring text coordinates. a* refer to SROIE from STNet*, b* refer to CORD from STNet*, and c* refer to DocVQA from STNet. For SROIE and CORD, different colors of the polygon boxes represent various categories. Each dataset has a different color bar, but both use green to indicate the ground truth. For DocVQA, different colors of the polygon boxes represent answers to different questions.

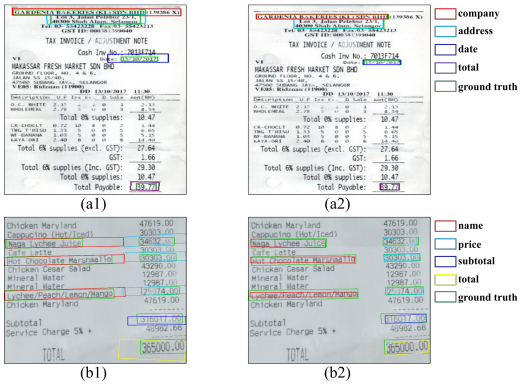


Figure 6: Comparison of text coordinates acquired by T3 and T4. a* refer to SROIE, b* refer to CORD. (a1) and (b1) refer to T3, (a2) and (b2) refer to T4. For clarity, some polygon boxes have been omitted.

Threshold	$1e^{-3}$	$1e^{-2}$	$1e^{-1}$	$3e^{-1}$
T3	86.3	85.5	80.6	58.7
T4	97.6	97.6	97.0	96.5

Table 5: Accuracy results for polygon box predictions. A prediction is considered correct if the IoU exceeds the defined threshold.

forms only slightly worse than T4. Figure 6 illustrates that while T3's predicted coordinates are less precise, they still encompass the correct answers. This indicates that our approach performs well even without precise location information. An IoU test on predicted boxes (Table 5) further demonstrates that T3 can locate the approximate position of the answer. Additionally, T4 provides more precise boxes for the answer locations, which correspond to its more accurate predictions of the answer text, validating the alignment between text and boxes.

7 Conclusion

In this work, we introduce STNet, a novel end-to-end model that not only provides textual answers but also excels in offering vision grounding. STNet employs a "see then tell" strategy, first outputting a special <see> token that encodes the answer's coordinates within the image as vision grounding to guide subsequent text generation. A dedicated physical decoder and a corresponding see loss are designed to decode and supervise these coordinates. To effectively train <see>, we collect a number of table recognition datasets and develop a GPT-4-driven automated QA pair generation method, resulting in the TVG dataset, which comprises QA pairs with precise vision grounding. Experimental results on publicly available datasets such as CORD, SROIE, and DocVQA demonstrate that our STNet model achieves state-of-the-art performance in Key Information Extraction.

8 Limitations

Our "see then tell" approach improves model performance on KIE tasks and achieves significant gains across multiple benchmarks. However, KIE is a relatively simple visual information extraction task, where inserting the <see> token at the end of the prompt or the beginning of the response already allows the model to utilize vision grounding effectively to generate the desired answer. In contrast, complex multi-step reasoning tasks require not only vision grounding but also strong textual inference capabilities. In such cases, how to enable the model to actively invoke visual perception by generating the <see> token at appropriate positions during reasoning remains an open problem for future exploration.

References

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R. Manmatha. 2021. Docformer: End-to-end transformer for document understanding. In *ICCV*, pages 973–983. IEEE.

Bernhard Axmann and Harmoko Harmoko. 2020. Robotic process automation: An overview and comparison to other technology in industry 4.0. In *ACIT*, pages 559–562. IEEE.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, and 12 others. 2020. Language models are few-shot learners. In *NeurIPS*.

Haoyu Cao, Changcun Bao, Chaohu Liu, Huang Chen, Kun Yin, Hao Liu, Yinsong Liu, Deqiang Jiang, and Xing Sun. 2023. Attention where it matters: Rethinking visual document understanding with selective region concentration. In *ICCV*, pages 19460–19470. IEEE.

Ting Chen, Saurabh Saxena, Lala Li, David J. Fleet, and Geoffrey E. Hinton. 2022. Pix2seq: A language modeling framework for object detection. In *ICLR*. OpenReview.net.

Lei Cui, Yiheng Xu, Tengchao Lv, and Furu Wei. 2021. Document AI: benchmarks, models and applications. *CoRR*, abs/2111.08609.

Yucheng Han, Chi Zhang, Xin Chen, Xu Yang, Zhibin Wang, Gang Yu, Bin Fu, and Hanwang Zhang. 2023. Chartllama: A multimodal LLM for chart understanding and generation. *CoRR*, abs/2311.16483.

Teakgyu Hong, Donghyun Kim, Mingi Ji, Wonseok Hwang, Daehyun Nam, and Sungrae Park. 2022. BROS: A pre-trained language model focusing on text and layout for better key information extraction from documents. In *Thirty-Sixth AAAI Conference on Artificial Intelligence, AAAI 2022, Thirty-Fourth Conference on Innovative Applications of Artificial Intelligence, IAAI 2022, The Twelveth Symposium on Educational Advances in Artificial Intelligence, EAAI 2022 Virtual Event, February 22 - March 1, 2022*, pages 10767–10775. AAAI.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. Layoutlmv3: Pre-training for document AI with unified text and image masking. In *ACM*, pages 4083–4091. ACM.

Zheng Huang, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and C. V. Jawahar. 2021. ICDAR2019 competition on scanned receipt OCR and information extraction. *CoRR*, abs/2103.10213.

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

Zi-Han Jiang, Chien-Wei Lin, Wei-Hua Li, Hsuan-Tung Liu, Yi-Ren Yeh, and Chu-Song Chen. 2025. Relation-rich visual document generator for visual information extraction. *arXiv preprint arXiv:2504.10659*.

Taeho Kil, Seonghyeon Kim, Sukmin Seo, Yoonsik Kim, and Daehee Kim. 2023. Towards unified scene text spotting based on sequence generation. In *CVPR*, pages 15223–15232. IEEE.

Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2022. Ocr-free document understanding transformer. In *ECCV*, volume 13688 of *Lecture Notes in Computer Science*, pages 498–517. Springer.

Diederik P. Kingma and Jimmy Ba. 2015. Adam: A method for stochastic optimization. In *ICLR*.

Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. 2023. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 18893–18912. PMLR.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Veselin Stoyanov, and Luke Zettlemoyer. 2020. BART: denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *ACL*, pages 7871–7880. ACL.

Chenxia Li, Weiwei Liu, Ruoyu Guo, Xiaoting Yin, Kaitao Jiang, Yongkun Du, Yuning Du, Lingfeng Zhu, Baohua Lai, Xiaoguang Hu, Dianhai Yu, and Yanjun Ma. 2022. Pp-ocrv3: More attempts for the improvement of ultra lightweight OCR system. *CoRR*, abs/2206.03001.

Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. 2021. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, pages 9992–10002. IEEE.

Pan Lu, Swaroop Mishra, Tanglin Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. 2022. Learn to explain: Multimodal reasoning via thought chains for science question answering. In *NeurIPS*.

Chuwei Luo, Yufan Shen, Zhaoqing Zhu, Qi Zheng, Zhi Yu, and Cong Yao. 2024. Layoutllm: Layout instruction tuning with large language models for document

understanding. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 15630–15640.

Yosi Mass and Haggai Roitman. 2020. Ad-hoc document retrieval using weak-supervision with BERT and GPT2. In *EMNLP*, pages 4191–4197. ACL.

Minesh Mathew, Dimosthenis Karatzas, and C. V. Jawahar. 2021. Docvqa: A dataset for VQA on document images. In *WACV*, pages 2199–2208. IEEE.

Yamato Okamoto, Youngmin Baek, Geewook Kim, Ryota Nakao, Donghyun Kim, Moonbin Yim, Seunghyun Park, and Bado Lee. 2024. CREPE: coordinate-aware end-to-end document parser. *CoRR*, abs/2405.00260.

OpenAI. 2023. GPT-4 technical report. *CoRR*, abs/2303.08774.

Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. 2019. Cord: a consolidated receipt dataset for post-ocr parsing. In *NeurIPS*.

Panupong Pasupat and Percy Liang. 2015. Compositional semantic parsing on semi-structured tables. In *ACL*, pages 1470–1480. ACL.

Baolin Peng, Chunyuan Li, Pengcheng He, Michel Galley, and Jianfeng Gao. 2023. Instruction tuning with GPT-4. *CoRR*, abs/2304.03277.

Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. 2020. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE.

Amanpreet Singh, Vivek Natarajan, Meet Shah, Yu Jiang, Xinlei Chen, Dhruv Batra, Devi Parikh, and Marcus Rohrbach. 2019. Towards VQA models that can read. In *CVPR*, pages 8317–8326. CVF / IEEE.

Brandon Smock, Rohith Pesala, and Robin Abraham. 2022. Pubtables-1m: Towards comprehensive table extraction from unstructured documents. In *CVPR*, pages 4624–4632. IEEE.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *NeurIPS*, pages 5998–6008.

Peng Wang, Shuai Bai, Sinan Tan, Shijie Wang, Zhihao Fan, Jinze Bai, Keqin Chen, Xuejing Liu, Jialin Wang, Wenbin Ge, and 1 others. 2024. Qwen2-vl: Enhancing vision-language model's perception of the world at any resolution. *arXiv preprint arXiv:2409.12191*.

Can Xu, Qingfeng Sun, Kai Zheng, Xiubo Geng, Pu Zhao, Jiazhan Feng, Chongyang Tao, and Daxin Jiang. 2023. Wizardlm: Empowering large language models to follow complex instructions. *CoRR*, abs/2304.12244.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei A. F. Florêncio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2021. Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. In *IJCNLP*, pages 2579–2591. Association for Computational Linguistics.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. Layoutlm: Pre-training of text and layout for document image understanding. In *ACM*, pages 1192–1200. ACM.

Tianshu Zhang, Xiang Yue, Yifei Li, and Huan Sun. 2023a. Tablellama: Towards open large generalist models for tables. *CoRR*, abs/2311.09206.

Zhenrong Zhang, Pengfei Hu, Jiefeng Ma, Jun Du, Jianshu Zhang, Baocai Yin, Bing Yin, and Cong Liu. 2024. Semv2: Table separation line detection based on instance segmentation. *PR*, 149:110279.

Zhenrong Zhang, Jiefeng Ma, Jun Du, Licheng Wang, and Jianshu Zhang. 2023b. Multimodal pre-training based on graph attention network for document understanding. *IEEE*, 25:6743–6755.

Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno-Yepes. 2020. Image-based table recognition: Data, model, and evaluation. In *ECCV*, volume 12366 of *Lecture Notes in Computer Science*, pages 564–580. Springer.

A Implementation Details

A.1 Model Architecture

Our proposed STNet utilizes specific hyperparameters: We set the input image resolution to 1280×960 and use random padding to maintain the original aspect ratio. The visual backbone's downsampling factor is configured to 32. The feature dimension D is established at 1024. The decoders consist of a stack of 4 identical layers, and the number of multi-heads is set to 16.

A.2 Pre-training Tasks

To bolster STNet's capability to perceive text locations — essentially, its ability to *see* — we have integrated a multi-task pre-training strategy encompassing three distinct sub-tasks: OCR, Document Read, and VQA, as illustrated in Figure 4.

A.2.1 OCR.

We represent locations using polygons defined by four coordinate points, with each point mapped to a token ranging from <0> to <999> in ***Loc***. STNet is designed to output text relevant to these locations based on such prompts, thereby emulating the process of an OCR engine.

A.2.2 Document Read.

This task improves the model's ability to understand document structures by training it to generate text sequences in the conventional reading order. Each text block in the sequence is prefaced by a <see> token. The physical coordinates of the text are obtained by decoding the hidden states using a specialized physical decoder.

A.2.3 VQA.

Expanding beyond conventional VQA tasks, STNet is designed to not only generate a plaintext response but also identify the relevant text coordinates using the <see> token for vision grounding. This approach aligns with the requirements of downstream KIE tasks.

A.3 Training Strategy

In the initial pre-training phase, in addition to the previously constructed TVG dataset and its data sources — the training sets from PubTables1M (Smock et al., 2022) and iFLYTAB (Zhang et al., 2024) — we also employ a synthetic dataset comprising 2.2 million entries in both Chinese and English from SynthDog (Kim et al., 2022). PubTables1M, iFLYTAB, and SynthDog are used for OCR and document read training, while TVG is utilized for OCR and VQA tasks.

After pre-training, STNet is fine-tuned on specialized datasets for KIE. Each dataset is tailored to meet the VQA task specifications and is combined with TVG at a 1:1 ratio, with consistent supervision of the *see* loss on TVG. It ensures that the output <see> token retains its physical location perception ability to guide the output of answer text, even without *see* loss supervision on downstream datasets.

We use the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 5×10^{-5}. The learning rate is linearly warmed up during the first 10% of the steps and then linearly decayed. The training is conducted on 4 Tesla V100 48GB GPUs with a total batch size of 28. The model is trained for 250 epochs on the SROIE (Huang et al., 2021) and CORD (Park et al., 2019) datasets, and extended to 300 epochs for DocVQA (Mathew et al., 2021).

STNet utilizes two types of loss: $\mathscr{L}_{\text{lm}}$ and $\mathscr{L}_{\text{see}}$. The total loss is computed as a weighted sum of these components.

$$\mathscr{L}_{\text{total}} = \mathscr{L}_{\text{lm}} + \lambda \mathscr{L}_{\text{see}} \tag{7}$$

After extensive evaluation, we set $\lambda = 0.001$.

B Experiments

B.1 Evaluation Benchmarks and Metrics

To fully demonstrate the effectiveness of STNet, we conduct experiments on three benchmark datasets.

B.1.1 SROIE.

The SROIE (Huang et al., 2021) dataset consists of 973 scanned receipt images. They are divided into two subsets: 626 images for training and 347 for testing. Each receipt is annotated with four predefined target fields: company, date, address, and total. Segment-level text bounding boxes and their corresponding transcripts are provided to facilitate the extraction tasks. The primary objective is to accurately map each word to its field. To achieve this, we have formulated four distinct queries, each addressing a specific target field: "What is the name of the company that issued this receipt?" for company,"Where was this receipt issued?" for address, "When was this receipt issued?" for date, and "What is the total amount on this receipt?" for total.

For evaluating model performance on the test set, we employ two metrics: the field-level F1 score (Hwang et al., 2019) and Tree Edit Distance (TED)-based accuracy (Zhong et al.,

Method	OCR	CORD	SROIE
		F1	F1
BROS (Hong et al., 2022)	✔	74.7	-
LayoutLM (Xu et al., 2020)	✔	78.4	-
LayoutLMv2 (Xu et al., 2021)	✔	78.9	61.0
LayoutLMv3 (Huang et al., 2022)	✔	80.5	65.0
BROS* (Hong et al., 2022)	✔	96.5	96.3
LayoutLM* (Xu et al., 2020)	✔	-	94.0
LayoutLMv2* (Xu et al., 2021)	✔	95.0	96.3
LayoutLMv3* (Huang et al., 2022)	✔	96.6	-
STNet	✘	88.1	87.8

Table 6: Comparison with OCR-based SOTA methods across different datasets. The field-level F1 scores are reported. Models marked with * utilize the ground truth of text strings and coordinates as inputs during evaluation.

2020). F1 score is a harmonic mean of precision and recall of a classification task. A high F1 score indicates strong performance in both accuracy and precision in classifying positive cases. However, it can not effectively reflect the prediction accuracy at the character level. TED measures the minimum number of single-character edit operations required to transform one string into another. The TED score is computed as $max(0, 1 - TED(pr, gt)/TED(\phi, gt))$, where gt represents the ground truth, pr denotes the predicted string, and ϕ corresponds to an empty string.

B.1.2 CORD.

The CORD (Park et al., 2019) dataset serves as a public benchmark comprising 800 training, 100 validation, and 100 testing receipts. The receipts are annotated with 30 types of entities under 4 categories: menu, void menu, subtotal, and total. A list of text lines with bounding boxes is provided. The evaluation task and metrics for the CORD dataset align with those used for the SROIE dataset. As it features an intricate nested structure, we use "parse the receipt" as a prompt, following the answer output format pioneered by Donut (Kim et al., 2022), and prepend a <see> token to each answer text.

B.1.3 DocVQA.

The DocVQA (Mathew et al., 2021) dataset comprises 50,000 Question Answering (QA) pairs derived from over 12,000 pages across a wide array of documents. The pages are allocated into training, validation, and test sets with an approximate ratio of 8:1:1. Due to the absence of ground truth in the test set, evaluations are performed on the validation set using the ANLS (Average Normalized Levenshtein Similarity), an edit-distance based metric.

B.2 Comparison with OCR-based Methods

OCR-based methods, such as the LayoutLM family (Xu et al., 2020, 2021; Huang et al., 2022), typically use the ground truth of text strings and coordinates during evaluation, as highlighted in their respective papers. This approach simplifies the task to a token classification problem based purely on textual content. To ensure a fair comparison, Donut (Kim et al., 2022) and SeRum (Cao et al., 2023) re-evaluated these models using state-of-the-art publicly available OCR engines to extract text and corresponding bounding boxes. Following this standard practice, we adopt the results reported in Donut and SeRum and re-evaluate LayoutLMv3 (Huang et al., 2022) under the same conditions.

As shown in Table 6, we compare the performance of these models under two evaluation settings. Notably, models marked with * utilize the ground truth of text strings and coordinates as inputs during evaluation.

It can be observed that these OCR-based methods can achieve satisfactory results when the OCR outputs are entirely accurate. However, producing such precise annotations is costly, and the cascading effects of OCR errors significantly impact the model's performance. In scenarios where only text and bounding boxes extracted by OCR engines are provided, our STNet demonstrates superior performance compared to these methods.

B.3 Generalization to Advanced MLLMs

To assess the generalizability of our approach, we apply it to advanced multimodal large language models (MLLMs) such as Qwen2-VL (Wang et al., 2024). We integrate our vision grounding module into the MLLM architecture and augment the prompt with the <see> token. The model is trained on the TVG dataset with explicit *see* supervision on <see> to acquire *see* capabilities. We adopt the Adam optimizer (Kingma and Ba, 2015) with a learning rate of 1×10^{-5}, which is linearly warmed up over the first 5% of training steps and then decayed linearly. Training is performed on 4 Tesla V100 GPUs (48GB each) with a total batch size of 128 for 3 epochs. To support full-parameter fine-tuning under memory constraints, we employ ZeRO optimization (Rajbhandari et al., 2020).

During evaluation, we follow the zero-shot prompt setting adopted in prior MLLM-based KIE studies (Luo et al., 2024). For SROIE and DocVQA, we adopt the same prompt format as used in STNet, with the <see> token appended at the end of the prompt. For key-value annotations in CORD, we format the queries as: "Q: What is the 'key'? <see> A: 'value'". To ensure a fair comparison, we follow previous work by filtering out samples where a single entity corresponds to multiple values. All evaluations are performed using the ANLS metric. Examples of the prompt formats and the text coordinates predicted by Qwen2-VL-2B with *see* are shown in Figure 7.

C TVG Dataset

As illustrated in Figure 3, we propose a comprehensive GPT-4-based method for the automatic construction of the TVG dataset. Each step of this process is detailed below.

C.1 Details of Data Source

Tables, as a unique form of document image, can be described using structured languages such as HTML. High-quality table data can be readily sourced from online resources. To this end, we have compiled several table recognition datasets, including PubTables1M (Smock et al., 2022) and iFLYTAB (Zhang et al., 2024).

C.1.1 PubTables1M.

PubTables1M is a large-scale table recognition dataset sourced from the PubMed Central Open Access (PMCOA) database. This dataset includes detailed annotations for projected row headers and bounding boxes for all rows, columns, and cells, including blank cells. Additionally, it introduces a novel canonicalization procedure aimed at correcting over-segmentation. This procedure ensures that each table is presented with a unique and unambiguous structural interpretation. Through these detailed annotations, we transform the tables into structured sequences in HTML format.

C.1.2 iFLYTAB.

The iFLYTAB dataset comprises 12,104 training samples and 5,187 testing samples. It offers comprehensive annotations for each table image, including both physical coordinates and detailed structural information. This dataset includes not only axis-aligned digital documents but also images captured by cameras, which present more significant challenges than PubTables1M due to their complex backgrounds and non-rigid image deformations. Although it lacks textual annotations, we have addressed this limitation by using PaddleOCR (Li et al., 2022) for text recognition, subsequently converting the tables into HTML format.

C.2 Generation Prompt

As shown in Figure 8, we present a standardized prompt template designed for QA data generation using GPT-4, which requires structured HTML table sequences as input. The text in black represents fixed components of the prompt, and the text within red brackets requires specific input. For example, $[Language]$ specifies the language in which the QA pairs should be generated. We instruct GPT-4 to generate five types of questions: specific extraction, simple reasoning, complex reasoning, numerical questions, and content summary.

C.2.1 Specific Extraction.

Each specific extraction question should target a specific cell in the table. The answer should indicate the row <tr> and column <td> of the cell.

C.2.2 Simple Reasoning.

Each simple reasoning question should have an answer derived by reasoning from fewer than three cells in the table.

C.2.3 Complex Reasoning.

Each complex reasoning question should have an answer that requires reasoning from three or more cells in the table.

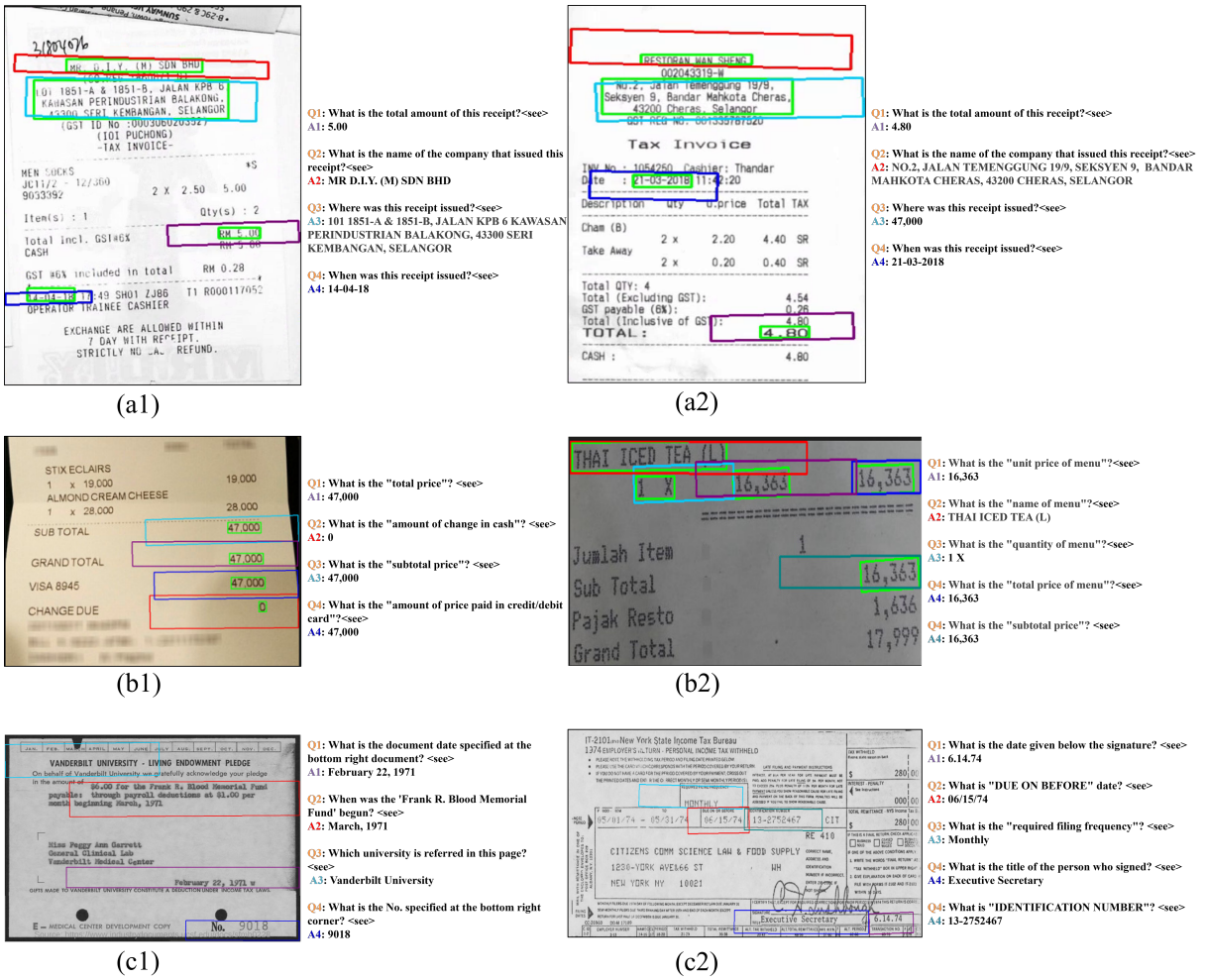


Figure 7: The results of acquiring text coordinates by Qwen2-VL-2B with *see*. a* refer to SROIE, b* refer to CORD, and c* refer to DocVQA. Different colors of the polygon boxes represent answers to different questions.

C.2.4 Numerical Questions.

Each numerical question should involve numerical calculations, such as sum, maximum, average, and minimum values. Provide the calculation process and the final result.

C.2.5 Content Summary.

Each content summary needs to provide a summary that describes the main content of the table and matches the table's content.

C.3 Post Process

As described in the aforementioned prompt template, for specific extraction questions, we require GPT-4 to provide not only the specific value from the cell but also the logical location of the cell, indicating its row and column numbers within the table. The detailed annotations in these table recognition datasets enable us to accurately locate the corresponding cell's real information, including its content and polygon box, based on the logical location. We only retain the QA pair when the value provided by GPT-4 matches the content of the located cell, with the cell polygon box serving as the required physical location $\boldsymbol{p}$. This process ensures the generation of high-quality QA data $\{\boldsymbol{Q}, \boldsymbol{A}, \boldsymbol{p}\}$. Ultimately, the TVG dataset we construct comprises 958,000 questions derived from 65,000 table images. It includes 244k specific extraction questions, 293k simple reasoning questions, 191k complex reasoning questions, 166k numerical questions, and 64k content summary. Some examples of them are illustrated in Figure 9 and released in supplementary material. The whole dataset will be made publicly available.

GPT-4 for QA Data Generation

Please generate a structured QA dataset based on the provided HTML table content. The dataset should include the following types of questions: specific extraction questions, simple reasoning questions, complex reasoning questions, and numerical calculation questions. When generating questions, adapt them to the language of the table content. If the table is in Chinese, all questions should be in Chinese; if the table is in English, all questions should be in English. For each category, please provide approximately 5 QA pairs and ensure the returned Python dictionary is complete. If a category is not applicable or cannot be generated based on the table content, please skip that category to maintain the quality of the dataset. Additionally, provide a content summary that captures the main content of the table and matches the table's content.

Below are the requirements for constructing the QA dataset, the detailed Python dictionary template, and the HTML table content:
1.Specific Extraction Questions: Each question should target a specific cell in the table. The answer should indicate the row <tr> and column <td> of the cell.
2.Simple Reasoning Questions: Each question should have an answer derived by reasoning from fewer than three cells in the table.
3.Complex Reasoning Questions: Each question should have an answer that requires reasoning from three or more cells in the table.
4.Numerical Calculation Questions: Each question should involve numerical calculations, such as sum, maximum, average, and minimum values. Provide the calculation process and the final result.
5.Content Summary: Provide a summary that describes the main content of the table and matches the table's content.
Ensure all QA pairs are precise and clearly indicate the row and column numbers as well as their specific content. Use the following Python dictionary structure to return the dataset:

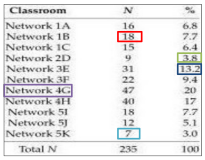
```
qa_dataset = {
  "specific_extraction": [
    {
      "question": "What is the data in <tr> X, <td> Y of the table?",
      "answer": {
        "tr": X,  # X must be int type
        "td": Y,  # Y must be int type
        "value": "Specific value from the cell"
      }
    },
    # ... Additional specific extraction questions in [Language]
  ],
  "simple_reasoning": [
    {
      "question": "What can be inferred about item A considering the data in cells B and C?",
      "answer": "Inferred answer based on reasoning"
    },
    # ... Additional simple reasoning questions in [Language]
  ],
  "complex_reasoning": [
    {
      "question": "Considering the data from cells X, Y, Z, what conclusion can we draw about the subject?",
      "answer": "Answer derived from complex reasoning"
    },
    # ... Additional complex reasoning questions in [Language]
  ],
  "numerical_questions": [
    {
      "question": "What is the sum of the values in column A?",
      "answer": "The numerical sum of column A, the final result is 1000"
    },
    # ... Additional numerical calculation questions in [Language]
  ],
  "content_summary": "The table primarily discusses topics related to [Topic], covering aspects like [Aspect 1], [Aspect 2], etc."
}
```

HTML table content:[HTML-formatted table]

Figure 8: The prompt template for QA data generation.

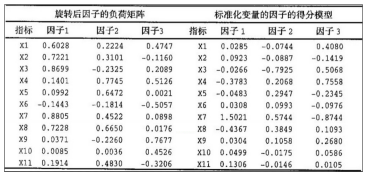
Q1: 2017年12月人事核数工资表中，第一个人的名字是什么?
A1: 赵飞燕
L1: 第2行，第2列
Q2: 2017年12月人事核数工资表中，高端发的前水是多少?
A2: 8004
L2: 第10行，第3列
Q3: 2017年12月人事核数工资表中，杨志超的合计工资是多少?
A3: 6,395.75
L3: 第4行，第11列
Q4: 2017年12月人事核数工资表中，何旭的医疗金是多少?
A4: 160.1
L4: 第12行，第5列
Q5: 2017年12月人事核数工资表中，润德轩的养老金是多少?
A5: 640.24
L5: 第8行，第4列

(a$_1$)



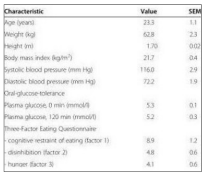
Q1: What is the number of students in Network 1B?
A1: 18
L1: 3rd <tr>, 2nd <td>
Q2: What is the percentage of Network 3E?
A2: 13.2
L2: 6th <tr>, 3rd <td>
Q3: What classroom is associated with a 20% representation?
A3: Network 4G
L3: 8th <tr>, 1st <td>
Q4: What is the number of students in Network 5K?
A4: 7
L4: 12th <tr>, 2nd <td>
Q5: What is the percentage for Network 2D?
A5: 3.8
L5: 5th <tr>, 3rd <td>

(a$_2$)



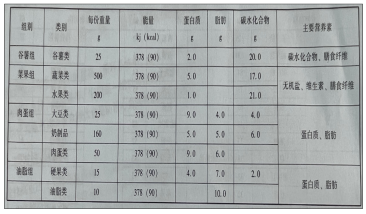
Q1: 在旋转后因子的负荷矩阵中，哪个指标在因子1上的负荷值最高?
A1: 指标X9的因子1上的负荷值最高。
Q2: 标准化变量的因子得分模型中，谁在因子2上有最负的得分?
A2: 指标X3在因子2上有最负的得分。
Q3: 旋转后因子的负荷矩阵中，是否有指标在所有三个因子上的负荷值都是正的?
A3: 没有，所有指标在至少一个因子上都有负的负荷值。
Q4: 在标准化变量的因子得分模型中，哪个指标在因子1上的得分最低?
A4: 指标X3在因子1上的得分最低。

(b$_1$)



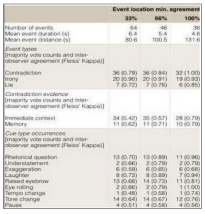
Q1: Does the participant have normal Systolic blood pressure?
A1: Yes, the Systolic blood pressure is 116.0 mm Hg which is within the normal range of 90–120 mm Hg.
Q2: Is the participant overweight based on the Body Mass Index value?
A2: No, the Body Mass Index is 21.7 which falls within the normal weight range (18.5–24.9).
Q3: Does the participant have normal blood glucose levels?
A3: Yes, the participant's blood glucose levels at 0 mins (5.3 mmol/l) and at 120 mins (5.2 mmol/l) are within the normal range (4.0–6.0 mmol/l).
Q4: Does this participant demonstrate restraint in eating?
A4: Yes, based on the Three-Factor Eating Questionnaire, the participant has a score of 8.9 in 'cognitive restraint of eating'.
Q5: Is the participant tall?
A5: The height of the participant is 1.70 meters, which is considered average.

(b$_2$)



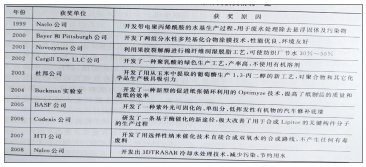
Q1: 如果我需要摄入蛋白质和脂肪，但又想控制碳水化合物的摄入量，我应该选择哪组类别的食品?
A1: 选择肉类蛋白和坚果类。蛋白含量高，脂肪适中，且碳水化合物较少。
Q2: 根据表格，如果想要高脂肪、高蛋白的食物，并避免过量的碳水化合物，我应该选择哪种类型的食物?
A2: 应该选择高蛋白食物（如豆类、肉类）以及富含高蛋白质（9.0g）和适当脂肪（4.0g）的食品。
Q3: 根据表格，如果想摄入低脂肪、碳水化合物，并适量摄入蛋白质，我应该如何选择食物?
A3: 选择适量的高纤维食物（如蔬菜类）和适量蛋白质的食物（如鸡蛋、豆类），这些食物能提供充足的蛋白质，而同时碳水化合物和脂肪含量较低。
Q4: 根据营养组的研究，如果每天从每组中选一种，我应该如何进行选择?
A4: 从高蛋白质含量的豆类和肉类中选择一种，同时从蔬菜类中选择一种，这样既能保证蛋白质摄入量，又能控制脂肪摄入量，从而减少碳水化合物和脂肪的摄入。
Q5: 根据表格，如果要控制高碳水化合物的饮食，应如何选择食物?
A5: 优先选择高蛋白质和富含维生素的蔬菜类食物（如蔬菜、花椰菜），避免高碳水化合物类食物（如米饭、面包）。

(c$_1$)



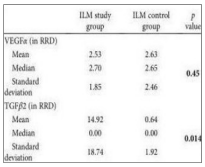
Q1: Considering the data for irony, lie, and contradiction, which type of event has the most consistent Fleiss' Kappa across different levels of agreement?
A1: Irony has the most consistent Fleiss' Kappa across different levels.
Q2: Based on the mean event duration and distance, how do the events change with the level of agreement?
A2: As the level of agreement increases, both the mean event duration and distance decrease.
Q3: Considering the observed cue types, which cue has the highest inter-observer agreement regardless of the event location agreement?
A3: Eye rolling has the highest inter-observer agreement.
Q4: Which evidence type has shown improvement in inter-observer agreement from 33% to 100% event location agreement?
A4: Both 'Immediate context' and 'Memory' have shown improvement in inter-observer agreement.
Q5: Taking into account all levels of event location agreement, which cue type has shown the greatest fluctuation in its Fleiss' Kappa score?
A5: Laughter has shown the greatest fluctuation in its Fleiss' Kappa score.

(c$_2$)



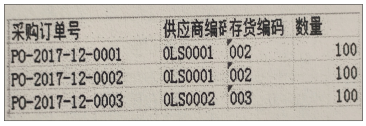
Q1: 从1999年到2008年，共有多少年颁发了奖项?
A1: 共有颁发了10年的奖项。
Q2: 2000年到2004年间，共有多少家公司获得了奖项?
A2: 共有5家公司获得了1次奖项。
Q3: Bayer公司和Pittsburgh公司在每一年中各获得了多少奖项?
A3: 在2000年中各获得了1次奖项。
Q4: 环保技术获奖的单位有多少个?
A4: 共有4个单位（包括Nalco公司、CargIl Dow LLC公司、Nalco公司和HTI公司）。
Q5: 获得可再生和节约用水技术的公司有哪三家?
A5: 共有3家公司（包括Nalco公司、Nalco公司和Novozymes公司）。

(d$_1$)



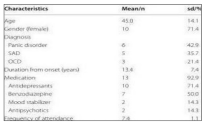
Q1: What is the sum of the mean values for VEGFα for both the ILM study and control groups?
A1: The sum of the mean values is 2.53 (study group) + 2.63 (control group) = 5.16.
Q2: What is the sum of the standard deviations for TGFβ2 for both the ILM study and control groups?
A2: The sum of the standard deviations is 18.74 (study group) + 1.92 (control group) = 20.66.
Q3: What is the mean value of the mean VEGFα for both groups?
A3: The mean value of the mean VEGFα is (2.53 + 2.63) / 2 = 2.58.
Q4: What is the difference in the mean values of TGFβ2 between the two groups?
A4: The difference in mean values is 14.92 (study group) - 0.64 (control group) = 14.28.
Q5: What is the difference in standard deviation for VEGFα between the two groups?
A5: The difference in standard deviation is 2.46 (control group) - 1.85 (study group) = 0.61.

(d$_2$)



Q1: 内容总结
A1: 表格主要记录了不同供应商编码下的采购订单信息，包含采购订单号、供应商编码、存货编码以及数量。共列出了三个订单，其中两个来自同一供应商编码OLS0001。

(e$_1$)



Q1: content_summary
A1: The table primarily discusses the details of a dataset concerning people's characteristics and their diagnoses of certain disorders. It includes age, gender, type of disorder, duration from onset, medication and frequency of attendance.

(e$_2$)

Figure 9: Some examples of the TVG dataset. a* refer to specific extraction, where L indicates the logical location and its text color corresponds to the coordinate box in the table image. b* refer to simple reasoning, c* refer to complex reasoning, d* refer to numerical questions, and e* refer to content summary.