PPTSER: A Plug-and-Play Tag-guided Method for Few-shot Semantic Entity Recognition on Visually-rich Documents

Anonymous ACL submission

Abstract

Visually-rich document information extraction (VIE) is a vital aspect of document understanding, wherein Semantic Entity Recognition (SER) plays a significant role. However, few-shot SER on visually-rich documents remains relatively unexplored despite its considerable potential for practical applications. To address this issue, we propose a simple yet effective **P**lug-and-**P**lay **T**ag-guided method for few-shot **S**emantic **E**ntity **R**ecognition (**PPTSER**) on visually-rich documents. PPTSER is built upon off-the-shelf multi-modal pre-trained models. It leverages the semantics of the tags to guide the SER task, reformulating SER into entity typing and span detection, handling both tasks simultaneously via cross-attention. Experimental results illustrate that PPTSER outperforms existing fine-tuning and few-shot methods, especially in low-data regimes. With full training data, PPTSER achieves comparable or superior performance to fine-tuning baseline. For instance, on the FUNSD benchmark, our method improves the performance of LayoutLMv3-base in 1-shot, 3-shot and 5-shot scenarios by 15.61%, 2.13%, and 2.01%, respectively. Overall, PPTSER demonstrates promising generalizability, effectiveness, and plug-and-play nature for few-shot SER on visually-rich documents. The codes will be available at https://anonymous.for.review.

1 Introduction

Information extraction from visually-rich documents (VIE) is a process that concentrates on extracting pertinent information from various sources such as scanned images, documents, and PDF files. It effectively leverages layout and visual cues to decode the content enclosed within these documents (Xu et al., 2020). As an important part of VIE, Semantic Entity Recognition (SER) aims to extract entity spans from the visually-rich document. SER has been hailed as a significant advancement in the realm of document intelligence, and has found widespread applications in numerous sectors.

The advent of multi-modal pre-trained models (Xu et al., 2020; Li et al., 2021c; Gu et al., 2021; Huang et al., 2022b; Yu et al., 2023) has ushered in a rapid evolution in SER methodologies. These models, pre-trained on a large corpus of scanned documents in a self-supervised manner, have significantly enhanced the comprehension ability of SER. Despite the remarkable achievements of the multi-modal pre-trained models, they often rely on extensive data for fine-tuning. However, acquiring a large volume of well-annotated SER data poses significant challenges such as: **(1)** Acquiring such data necessitates substantial financial resources and time. Annotators are required to label a multitude of OCR detection boxes in the document, adhering to meticulously designed guidelines. Identification of content within a box and accurately assigning labels to them are also tedious tasks. **(2)** The availability of data is often restricted due to privacy concerns. In scenarios involving sensitive information, such as invoices and insurance documents, data accessibility is severely limited due to the confidential nature of this information.

Despite the scarce research (Cheng et al., 2020; Yao et al., 2021; Wang and Shang, 2022) on few-shot Semantic Entity Recognition for visually-rich documents (few-shot SER), results have shown limitations in terms of generality and performance, and were limited to the specific application scenario. This paper, inspired by the comprehension capabilities of pre-trained models and the selective focus nature of the attention mechanism, introduces a novel approach called PPTSER, a **P**lug-and-**P**lay **T**ag-guided method for few-shot **S**emantic **E**ntity **R**ecognition on visually-rich documents. The underlying principle of PPTSER consists of three main components: **(1) Semantic Understanding and Alignment:** Words related to SER tags are used as a prompt and are concatenated with the

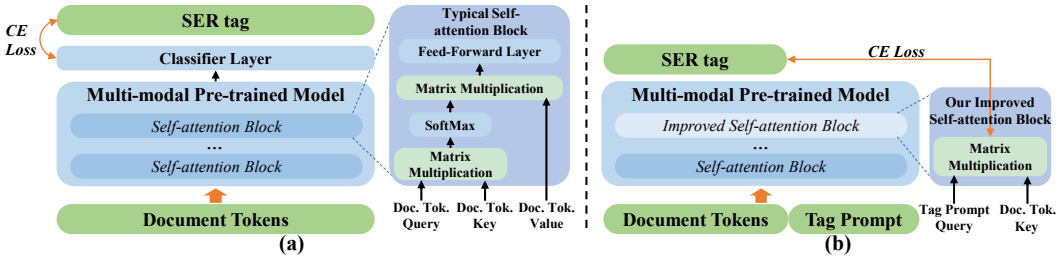

Figure 1: **(a)** Illustration of the traditional fine-tuning method. *Doc. Tok.* refers to *Document Tokens*. **(b)** Overview of our PPTSER method. PPTSER replaces the last self-attention block with an improved attention block and omits an extra classifier layer compared to traditional fine-tuning, which has less modules and parameters.

document's text tokens. This combined input is then fed into a multi-modal pre-trained model. The motivation behind this is that the pre-trained model is expected to understand the semantics of both the document tokens and the tag-related prompt, thereby bringing the hidden states of the tokens and tag-related words for a specific entity type closer together. **(2) Decoupling of SER task:** SER task is segmented into Entity Classification and Entity Boundary Detection. This division aims to facilitate the resolution of boundary determination among adjacent entities of the same category within visually-rich documents. **(3) Efficient Usage of Multi-head Attention:** The attention weight obtained from the last attention block between the tag-related prompt and document tokens is directly used as the probability of tokens belonging to different tags. This mechanism, with different heads detecting various spans, is ideal for the SER task with numerous entity spans. By fully exploiting the weighted focus nature of the attention mechanism, the model eliminates the value transform layer, feed-forward layer in the last attention block, and omits a separate classifier layer compared to traditional fine-tuning methods (as depicted in Figure 1), As a result, the total parameter is reduced.

Extensive experiments are conducted to show the PPTSER's effectiveness on commonly-used SER benchmarks, covering multiple languages, in few-shot to the full training set settings, and using different mainstream multi-modal pre-trained models.

The main contributions of this paper can be summarized as follows:

- We have demonstrated that the semantics of labels can effectively guide the SER task and have proposed a plug-and-play method ideal for few-shot SER on visually-rich documents. To the best of our knowledge, we are the first to propose a pluggable method that has shown effectiveness on various pre-trained models and languages.
- By innovatively leveraging the multi-head attention mechanism embedded in the pre-trained model, our method successfully extracts dense entities on visually-rich documents without adding any additional parameters.
- Experimental results show the superiority of our method over the traditional fine-tuning approaches in both few-shot and full-training-set scenarios. Moreover, PPTSER outperforms existing few-shot SER methods by significant margin, thereby underscoring its overall efficacy.

2 Method

2.1 Task Formulation

SER is usually formulated as a sequence labeling task. For given tokens from the document $\boldsymbol{x} = [x_i], i = 1, 2, ..., n$, SER aims to assign a label $y_i \in \mathbb{C}$ for each token x_i, where $\mathbb{C}$ is the SER label space. Subsequently, entity spans would be analyzed from the labeled tokens according to a specific scheme, such as `BIO` (Ramshaw and Marcus, 1995) and `IO` (Tjong Kim Sang and De Meulder, 2003).

In this paper, we primarily focus on the **In-Label-Space** setting for few-shot SER. Specifically, the pre-trained model is firstly fine-tuned on a small number of M annotated documents with label space $\mathbb{C}$ and then directly evaluated on the test set with the same label space $\mathbb{C}$. This task presents a significant challenge as the model needs to learn the SER task with only limited training samples.

It is notable that in the context of few-shot SER, the few-shot setting of *N-way K-shot* indicates that *each of the N categories has K documents containing entities of that category as the support set*, as visually-rich documents are annotated at document level. Moreover, a document often contains entity spans of distinct types, causing potential overlaps

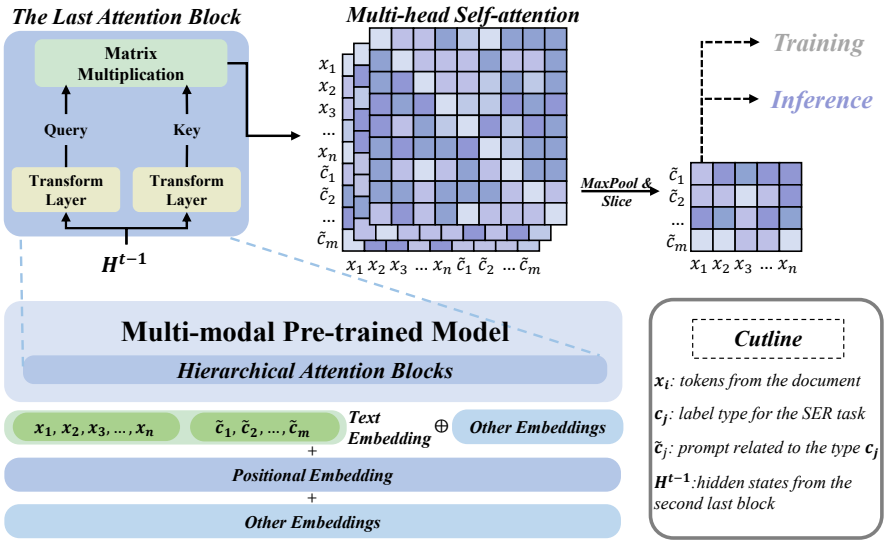

Figure 2: The overall architecture of PPTSER. *Other Embeddings* may include various embeddings such as *Visual Embedding*, *Type Embedding*, among others, with their presence and format dependent on the type of the pre-trained model used. In this architecture, the tokens extracted from documents and the tag-related prompt are concatenated and subsequently encoded with the pre-trained model. The attention weight, obtained from the last attention block between tokens and the prompt, is then used to ascertain whether the tokens correspond to the respective label type.

between the support sets for different entity types across N categories. Consequently, the overall number of annotated documents $M < N \times K$.

2.2 PPTSER

The fundamental concept and flow chart of PPTSER is shown in Figure 2. The method begins with the construction of a prompt based on SER tags. This prompt is then concatenated with the document tokens and jointly encoded using a unified pre-trained model. Within the transformer architecture of our model, attention weights between document tokens and the tag-related prompt are computed in hierarchical attention blocks. We use the attention weight between the tag-related prompt and document tokens, which can be considered as a form of cross-attention, obtained from the last attention block as the probability distribution of tokens belonging to different SER entity types.

2.2.1 Tag-related Prompt Construction and Target Generation

For an SER task with the label space $\mathbb{C}$, we need to construct tag-related words $\tilde{c}_i$ for each $c_i \in \mathbb{C}$, and then the tag-related prompt $\tilde{\mathbb{C}} = \{\tilde{c}_i\}, i = 1, 2, ...m$ is built. In PPTSER, we simply use the tag names as the tag-related words.

To enable PPTSER to accurately identify the boundaries of entity spans, we employ BIO tagging scheme in our method. However, when dealing with an SER task involving entity types $\mathbb{E} = \{e_i | e_0 = Other\}, i = 0, 1, 2, ..., m$ (where *Other* represents the entities that are not of interest), the label space would be $\mathbb{C} = \{e_0, B_{e_i}, I_{e_i}\}$, and the prompt would be $\tilde{\mathbb{C}} = \{e_0, beginning\, of\, e_i, inner\, of\, e_i\}$, where $i = 1, 2, ..., m$. In such a scenario, the prompt $\tilde{\mathbb{C}}$ becomes not only semantically redundant but also excessively long, potentially impeding the effective semantic learning of the document tokens.

Thus, we reframe the SER task with a BIO tagging scheme into two separate tasks: *entity typing* and *span detection*. Entity typing involves assigning an entity type for each document token, while span detection aims to identify whether tokens are at the beginning or interior of an entity span.

To further clarify, consider an SER task using BIO tagging scheme with a predefined entity type set $\mathbb{E} = \{e_i | e_0 = Other\}, i = 0, 1, 2, ..., m$. For entity typing, the label space and the tag-related prompt would be $\mathbb{C}^{ent.} = \{c_i^{ent.} | c_i^{ent.} = e_i\}$ and $\tilde{\mathbb{C}}^{ent.} = \{\tilde{c}_i^{ent.} | \tilde{c}_i^{ent.} = c_i^{ent.}\}$,where $i = 0, 1, 2, ..., m$; And for span detection, the label space and the prompt would be $\mathbb{C}^{det.} = \{c_1^{det.}, c_2^{det.}\}$ and $\tilde{\mathbb{C}}^{det.} = \{\tilde{c}_1^{det.}, \tilde{c}_2^{det.}\}$, where $\mathbb{C}^{det.} = \tilde{\mathbb{C}}^{det.} = \{beginning, inner\}$; Then the full label space and prompt would be $\mathbb{C} = \mathbb{C}^{ent.} \cup \mathbb{C}^{det.} = \{c_i^{ent.}, c_j^{det.}\}$ and $\tilde{\mathbb{C}} = \tilde{\mathbb{C}}^{ent.} \cup \tilde{\mathbb{C}}^{det.} = \{\tilde{c}_i^{ent.}, \tilde{c}_j^{det.}\}$, where $i = 0, 1, 2, ..., m$; $j = 1, 2$. For a token with an entity type of $e_i (i \neq 0)$ located at the $beginning/inner$ of an entity span, the corresponding labels would be c_i for entity typing and $beginning/inner$ for span detection. However,

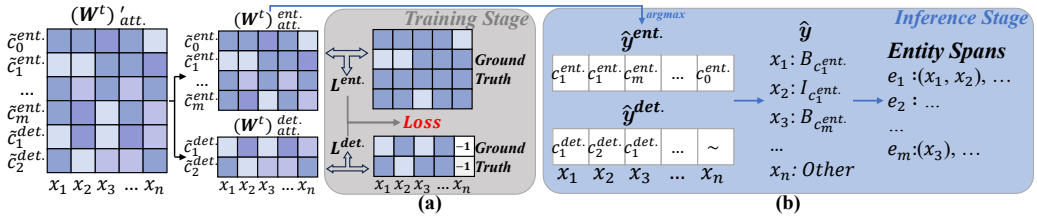

Figure 3: **(a)** PPTSER at training stage. Losses of entity typing and span detection are computed separately and then combined for the overall loss calculation. And -1 signifies that the loss at those points is disregarded. **(b)** PPTSER at inference stage. Combined predictions of entity typing and span detection are utilized to analyze the entity spans.

for the token with an entity type of $Other$, the specific location of it within an entity span is irrelevant, and the loss for span detection is ignored here. Consequently, we can formulate the entity typing target $\boldsymbol{y}^{ent.} = [y_i^{ent.}]$ and the span detection target $\boldsymbol{y}^{det.} = [y_i^{det.}]$, where $i = 1, 2, ..., n$.

It is worth emphasizing that our PPTSER framework handles entity typing and span detection simultaneously. And prompts for them $\tilde{\mathbb{C}}^{ent.}$ and $\tilde{\mathbb{C}}^{det.}$ are encoded in parallel, allowing them to benefit from each other during the learning process.

2.2.2 Cross-attention within the Pre-trained Model

Once the tag-related prompt $\tilde{\mathbb{C}}$ is constructed, it is concatenated with the document tokens $\boldsymbol{x} = [x_i], i = 1, 2, ..., n$, forming a boosted input $\boldsymbol{x}' = \boldsymbol{x} \bigoplus \tilde{\mathbb{C}} = [x_i, \tilde{c}_j^{ent.}, \tilde{c}_k^{det.}], i = 1, 2, ..., n;\ j = 0, 1, 2, ..., m;\ k = 1, 2$. Then, $\boldsymbol{x}'$ is used as the *Text Embedding* encoded in the pre-trained model. Notably, when $\tilde{\mathbb{C}}$ involves other kinds of embedding, such as positional or visual embeddings, they are set to 0, since $\tilde{\mathbb{C}}$ are hypothetical tokens not found in the document. Let's denote the hidden states from the second last block as $\boldsymbol{H}^{t-1}$:

$$\boldsymbol{H}^{t-1} = [\boldsymbol{h}_i^{t-1}, \tilde{\boldsymbol{h}}_j^{t-1}, \tilde{\boldsymbol{h}}_k^{t-1}] \tag{1}$$

where $\boldsymbol{h}_i^{t-1}, \tilde{\boldsymbol{h}}_j^{t-1}, \tilde{\boldsymbol{h}}_k^{t-1}$ are the hidden states for $\boldsymbol{x}, \tilde{\mathbb{C}}^{ent.}, \tilde{\mathbb{C}}^{det.}$, correspondingly.

Then, $\boldsymbol{H}^{t-1}$ is partitioned into multiple segments $\boldsymbol{H}_i^{t-1}$ along the channel dimension, where queries $\boldsymbol{Q}_i^t$ and keys $\boldsymbol{K}_i^t$ of the ith attention head are transformed as follows:

$$\boldsymbol{Q}_i^t = (W_i^t)_q \boldsymbol{H}_i^{t-1} \tag{2}$$

$$\boldsymbol{K}_i^t = (W_i^t)_k \boldsymbol{H}_i^{t-1} \tag{3}$$

where $(W_i^t)_q$ and $(W_i^t)_k$ are learnable weights embedded in the last attention block. And the self-attention weight of distinct heads is computed as below:

$$(\boldsymbol{W}_i^t)_{att.} = \boldsymbol{Q}_i^t (\boldsymbol{K}_i^t)^T \tag{4}$$

where $(\boldsymbol{W}_i^t)_{att.}$ is a matrix with the shape of $(n + m + 3) \times (n + m + 3)$. From this matrix, we extract a sub-matrix $(\boldsymbol{W}_i^t)'_{att.}$ that takes the prompt as queries and the document tokens as keys, which possesses the shape of $(m+3) \times n$. $(\boldsymbol{W}_i^t)'_{att.}$ can be viewed as a form of cross-attention within the self-attention, which depicts the relationship between the tag-related prompt and document tokens.

We hypothesize that distinct heads of the attention mechanism enable the prompt to focus on distinct entity spans, which is suitable for the entity-rich scenario in visually-rich documents. We select the maximum weight across heads to get a summary relationship between the prompt and tokens:

$$(\boldsymbol{W}^t)'_{att.} = \max_{i \in \{1,2,...,l\}} (\boldsymbol{W}_i^t)'_{att.} \tag{5}$$

Further, $(\boldsymbol{W}^t)'_{att.}$ is partitioned into two components, namely $(\boldsymbol{W}^t)^{ent.}_{att.}$ and $(\boldsymbol{W}^t)^{det.}_{att.}$ as shown in Figure 3(a). These components use the hidden states of $\tilde{\mathbb{C}}^{ent.}$ and $\tilde{\mathbb{C}}^{det.}$ as queries, and possess the shape of $(m + 1) \times n$ and $2 \times n$, correspondingly. $(\boldsymbol{W}^t)^{ent.}_{att.}$ and $(\boldsymbol{W}^t)^{det.}_{att.}$ represent the probability distribution for document tokens belonging to distinct tags. The losses are then calculated as follows:

$$L^{ent.} = -\frac{1}{n} \sum_{i=1}^{n} \frac{exp(w_{pi}^{ent.})}{\sum_{j=0}^{m} exp(w_{ji}^{ent.})} \tag{6}$$

$$L^{det.} = -\frac{1}{n} \sum_{i=1}^{n} \frac{exp(w_{qi}^{det.})}{\sum_{j=1}^{2} exp(w_{ji}^{det.})} \tag{7}$$

where $L^{ent.}$ and $L^{det.}$ are the losses for entity typing and span detection, $w_{ij}^{ent.}$ and $w_{ij}^{det.}$ are elements in $(\boldsymbol{W}^t)^{ent.}_{att.}$ and $(\boldsymbol{W}^t)^{det.}_{att.}$, and $y_i^{ent.} = c_p^{ent.}$, $y_i^{det.} = c_q^{det.}$. And the total loss is formulated as follows:

$$Loss = L^{ent.} + \alpha L^{det.} \tag{8}$$

Here, α is the ratio factor to balance the losses, and we set $\alpha = 0.1$ for models with segment-level positional embeddings and $\alpha = 1.5$ for models with word-level positional embeddings.

Modality		Text + Layout				Text + Layout + Image			
Methodology		BROS (AAAI 22)		LiLT (ACL 22)		LayoutLMv2 (ACL 21)		LayoutLMv3 (MM 22)	
		FT	**Ours**	FT	**Ours**	FT	**Ours**	FT	**Ours**
FUNSD	1-shot	48.08	**54.39** $^{\uparrow 6.31}$	52.60	**55.64** $^{\uparrow 3.04}$	48.22	**52.17** $^{\uparrow 3.95}$	46.37	**61.98** $^{\uparrow 15.61}$
	3-shot	64.34	**67.70** $^{\uparrow 3.36}$	67.64	**69.17** $^{\uparrow 1.52}$	61.66	**63.64** $^{\uparrow 1.98}$	74.73	**76.86** $^{\uparrow 2.13}$
	5-shot	67.77	**70.64** $^{\uparrow 2.87}$	73.29	**75.26** $^{\uparrow 1.97}$	65.86	**67.49** $^{\uparrow 1.63}$	79.52	**81.53** $^{\uparrow 2.01}$
	7-shot	68.21	**71.96** $^{\uparrow 3.75}$	73.39	**75.71** $^{\uparrow 2.32}$	66.55	**68.83** $^{\uparrow 2.28}$	79.84	**81.60** $^{\uparrow 1.76}$
	Full Data	83.83	**83.91** $^{\uparrow 0.08}$	88.95	**89.07** $^{\uparrow 0.12}$	83.52	**83.72** $^{\uparrow 0.20}$	91.15	**92.01** $^{\uparrow 0.86}$
CORD	1-shot	66.28	**68.48** $^{\uparrow 2.20}$	70.04	**75.57** $^{\uparrow 5.54}$	69.61	**69.97** $^{\uparrow 0.36}$	70.35	**74.19** $^{\uparrow 3.84}$
	3-shot	79.02	**79.61** $^{\uparrow 0.59}$	81.64	**83.83** $^{\uparrow 2.19}$	80.63	**81.66** $^{\uparrow 1.03}$	82.05	**85.27** $^{\uparrow 3.22}$
	5-shot	84.04	**84.37** $^{\uparrow 0.34}$	85.52	**87.06** $^{\uparrow 1.54}$	84.32	**84.53** $^{\uparrow 0.21}$	85.83	**87.77** $^{\uparrow 1.94}$
	7-shot	83.68	**84.09** $^{\uparrow 0.42}$	85.35	**87.73** $^{\uparrow 2.38}$	84.76	**85.31** $^{\uparrow 0.55}$	86.94	**88.48** $^{\uparrow 1.54}$
	Full Data	95.72	**95.75** $^{\uparrow 0.03}$	95.80	**96.04** $^{\uparrow 0.25}$	95.20	**95.63** $^{\uparrow 0.44}$	96.34	**96.39** $^{\uparrow 0.05}$
XFUND-zh	1-shot	-	-	60.10	**67.64** $^{\uparrow 7.54}$	60.28	**68.26** $^{\uparrow 7.98}$	52.92	**56.65** $^{\uparrow 3.73}$
	3-shot	-	-	72.61	**74.17** $^{\uparrow 1.56}$	74.37	**77.20** $^{\uparrow 2.83}$	69.08	**75.24** $^{\uparrow 6.16}$
	5-shot	-	-	77.40	**79.40** $^{\uparrow 2.00}$	81.43	**82.34** $^{\uparrow 0.91}$	75.25	**79.26** $^{\uparrow 4.01}$
	7-shot	-	-	80.47	**81.38** $^{\uparrow 0.91}$	82.25	**83.66** $^{\uparrow 1.41}$	77.85	**80.97** $^{\uparrow 3.12}$
	Full Data	-	-	90.47	**90.61** $^{\uparrow 0.14}$	90.25	**90.79** $^{\uparrow 0.54}$	91.61	**92.19** $^{\uparrow 0.58}$

Table 1: F1 score (%) of PPTSER and traditional Fine-tuning methods. F1 score in **Bold** is better between our PPTSER and Fine-tuning. *FT* refers to *Fine-tuning* methods.

2.2.3 Decoding during the Inference Stage

The inference stage is shown in Figure 3(b). We first apply the $argmax$ operation on $(\boldsymbol{W}^t)^{ent.}_{att.}$ and $(\boldsymbol{W}^t)^{det.}_{att.}$ along distinct prompt words to get the predicted tag with the highest probability:

$$\hat{y}_i^{ent.} = \underset{j \in \{0,1,2...,m\}}{\operatorname{argmax}} \; w_{ji}^{ent.} \tag{9}$$

$$\hat{y}_i^{det.} = \underset{j \in \{1,2\}}{\operatorname{argmax}} \, w_{ji}^{det.} \tag{10}$$

Then the prediction with BIO tagging scheme $\hat{\boldsymbol{y}} = [\hat{y}_i], i = 1, 2, ..., n$ is formulated as follows:

$$\hat{y}_i = \begin{cases} B_{\hat{y}_i^{ent.}} & , \hat{y}_i^{ent.} \neq Other, \; \hat{y}_i^{det.} = beginning \\ I_{\hat{y}_i^{ent.}} & , \hat{y}_i^{ent.} \neq Other, \; \hat{y}_i^{det.} = inner \\ Other & , \hat{y}_i^{ent.} = Other \end{cases} \tag{11}$$

Finally, the entity spans are analyzed from $\hat{\boldsymbol{y}}$ using the BIO tagging scheme. Notably, spans not conform to BIO scheme, especially those starting with a token predicted as $\hat{y}_i^{det.} = inner$, are labeled as $Other$. This operation, aimed at enhancing predicting accuracy, is applied in both PPTSER and methods we compared to for a fair comparison.

To provide a more vivid demonstration of our method, we offer an example run with PPTSER, refer to Appendix C due to paper-length constraints.

3 Experiments

3.1 Experimental Settings

Benchmarks. We conducted experiments on several widely used SER benchmarks, including FUNSD (Jaume et al., 2019), CORD (Park et al., 2019) and XFUND (Xu et al., 2022). FUNSD targets form understanding with 199 scanned documents related to market reports, commercials, and more. CORD, centered on receipt understanding, features both coarse (e.g., *menu*, *total*) and fine-grained (e.g., *menu.unitprice*, *menu.price*) annotations. This benchmark provides an official split of training, validation and test sets, and we strictly follow the procedure by selecting the model weight that achieved the best performance on the validation set for testing on the test set. XFUND focuses on document understanding covering multiple languages. In this article, our primary focus is on the Chinese subset of XFUND, denoted as XFUND-zh.

Few-shot Settings. PPTSER was evaluated on 1-shot, 3-shot, 5-shot, 7-shot and the full training set scenarios. With no official few-shot divisions in benchmarks mentioned above, we established our own following the process in Appendix A. We selected as few samples as possible while meeting the few-shot setting, which aligns with the real-world application. Due to the inherent instability of few-shot experiments, we randomly generated 5 different divisions for every scenario and tested each with 2 diverse random seeds. Hence, our experiment result is the average of 10 runs, ensuring the reliability and credibility of our findings.

3.2 Comparisons with Existing Fine-Tuning Methods

Setup. The foundation for our method is built upon several widely used multi-modal pre-trained models, incorporating different combinations of modalities as input. This includes **BROS** (Hong et al., 2022) and **LiLT** (Wang et al., 2022a) with textual and layout input, and **LayoutLMv2** (Xu et al.,

Modality		Text		Text + Layout			Text + Layout + Image	
Methodology		EntLM (NAACL 22)	COPNER (COLING 22)	LASER (ACL 22)	COPNER$_{LiLT}$ (COLING 22)	**PPTSER$_{LiLT}$ (Ours)**	COPNER$_{LMv3}$ (COLING 22)	**PPTSER$_{LMv3}$ (Ours)**
FUNSD	1-shot	24.32	19.37	38.47	55.15	55.64	51.19	**61.98**
	3-shot	34.94	31.21	44.88	68.66	69.17	75.84	**76.86**
	5-shot	39.55	35.13	49.31	73.43	75.26	77.55	**81.53**
	7-shot	41.41	37.31	52.56	73.35	75.71	78.53	**81.60**
	Full Data	67.42	64.58	69.23	87.74	89.07	91.26	**92.01**
CORD-Lv1	1-shot	74.29	68.61	66.80	86.97	**90.50**	86.98	90.02
	3-shot	83.68	82.25	76.09	94.16	94.79	94.03	**95.13**
	5-shot	87.11	86.08	82.23	94.86	**96.21**	95.74	**96.21**
	7-shot	87.31	86.74	83.61	95.04	96.13	96.06	**96.51**
	Full Data	95.93	95.90	96.56	99.21	99.42	**99.45**	**99.45**
CORD	1-shot	57.86	54.52	-	70.05	**75.57**	67.33	74.19
	3-shot	71.68	71.32	-	81.27	83.83	80.07	**85.27**
	5-shot	77.74	78.98	-	84.80	87.06	85.30	**87.77**
	7-shot	78.63	78.63	-	85.76	87.73	86.87	**88.48**
	Full Data	93.50	94.16	-	95.74	96.04	95.79	**96.39**
XFUND-zh	1-shot	26.38	23.29	-	48.76	**67.64**	54.26	56.71
	3-shot	37.22	37.49	-	64.59	74.17	71.27	**75.24**
	5-shot	43.54	44.36	-	69.03	**79.40**	76.37	79.26
	7-shot	46.62	46.90	-	74.44	**81.38**	79.29	80.97
	Full Data	66.20	67.11	-	89.17	90.61	91.99	**92.19**

Table 2: F1 score (%) of PPTSER and other Few-shot methods. F1 score in **Bold** is the best, and that with underline is the second best.

2021a) and **LayoutLMv3** (Huang et al., 2022b) with textual, layout and image input. Since BROS only supports English, we only tested it on FUNSD and CORD. For testing on XFUND-zh, we used **LayoutXLM** (Xu et al., 2021b), which is the multilingual version of LayoutLMv2. In our experiments, we utilized base-size pre-trained models.

Results. Table 1 showcases the performance of PPTSER against traditional fine-tuning methods. The results clearly demonstrate that our PPTSER outperforms traditional fine-tuning methods across all tested scenarios and benchmarks. This underscores the superior performance of PPTSER in diverse language contexts with various base models.

Overall, both PPTSER and the fine-tuning method demonstrate improved performance with increased training data. However, our PPTSER consistently outperforms previous fine-tuning methods in all few-shot settings, especially with exceptionally scarce data. In the 1-shot scenario on FUNSD, where only a single annotated document is available, PPTSER achieves gains of **+6.31%** with BROS, **+3.04%** with LiLT, **+3.95%** with LayoutLMv2 and the highest gain of **+15.62%** with LayoutLMv3, emphasizing its effectiveness in data-scarce situations. Notably, even when trained with the full training data, our PPTSER still achieves comparable performance to the fine-tuning method, and even outperforms it in certain scenarios. For example, we observe a gain of **+0.86%** on FUNSD with LayoutLMv3. This full data setting is often neglected in other few-shot research, further underscoring the superiority of our approach when dealing with varying amounts of available data.

Our findings demonstrate that PPTSER is highly adaptive to different amounts of training data with distinct base models, making it an effective method for addressing the SER problem.

3.3 Comparisons with Existing Few-shot Methods

Setup. We selected the PPTSER models that performed better under different modality settings, denoted as **PPTSER**$_{LiLT}$ and **PPTSER**$_{LMv3}$, and compared them with previous few-shot methods. For a comprehensive comparison, we re-implemented **LASER** (Wang and Shang, 2022) on our few-shot divisions. However, it can only handle the coarse-level typing for CORD (CORD-Lv1) and is limited to the English language. Since research on few-shot SER is rather limited, we selected two other few-shot NER methods for comparison. Specifically, We chose **COPNER** (Huang et al., 2022a) and **EntLM** (Ma et al., 2022b) due to their similar In-Label-Space setting with ours. Considering COPNER can also be used as a pluggable method, we also integrated it with LayoutLMv3 and LiLT, denoted as **COPNER**$_{LiLT}$ and **COPNER**$_{LMv3}$.

Results. The overall experimental results are presented in Table 2. The results clearly show that PPTSER outshines existing few-shot NER and SER methods by a large margin. Interestingly, COPNER shows some degree of pluggability with various multi-modal pre-trained models, but PPTSER still

Model Designs	**Baseline** *default setting*	**Decoupling Strategies** *plain BIO prompt*	**Prompt Engineering** *unrelated words*	*random embeddings*	**Aggregation Stratergies** *mean*	*single head*
1-shot	75.57	74.28	70.80	71.43	74.43	75.21
3-shot	83.83	82.41	82.69	83.07	83.05	83.92
5-shot	87.06	86.25	85.78	86.29	86.64	86.98
7-shot	87.73	86.49	86.42	86.95	86.74	87.24
Full Data	96.04	95.29	96.26	95.72	96.21	96.06
Δ	-	**-1.10**	**-1.66**	**-1.35**	**-0.63**	**-0.17**

Table 3: F1 score (%) of PPTSER on CORD benchmark with different designs. Δ denotes the average deviation of the F1 score relative to the *default setting*.

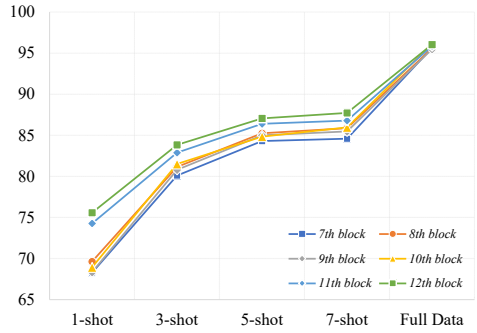

Figure 4: F1 score (%) of PPTSER on CORD benchmark with different settings when obtaining the attention weight from different blocks.

outperforms it across all settings and benchmarks.

In summary, our PPTSER surpasses existing few-shot NER and few-shot SER methods on various visually-rich documents, showcasing its effectiveness in handling few-shot SER challenge.

4 Ablation Study

We have conducted extensive analyses of our PPTSER to ensure its effectiveness and rationality. For convenience, experiments are conducted on the CORD dataset using PPTSER building upon LiLT.

Origin of Attention Weights. To pinpoint the source of superiority in PPTSER, we explored whether it stems from our meticulous design or the decreased over-fitting achieved through parameter reduction. We extracted attention weights from various blocks, including the default 12th block and shallower $7^{th} \sim 11^{th}$ block. And the experimental results shown in Figure 4 reveal that extracting attention weights from the last block is more effective than from other blocks, which has greatly assured the effectiveness of our design.

Effectiveness of Decoupling Strategies. Table 3 also shows comparisons with different frameworks of prompts. In this context, *default setting* refers to our design to decouple the SER task into entity typing and span detection then processing them concurrently, while *plain BIO prompt* refers to the direct usage of the aforementioned $\tilde{C} = \{e_0, beginning\, of\, e_i, inner\, of\, e_i\}$ as the prompt, without decoupling. The result shows that our decoupling avoids disrupting the language modeling of document tokens and performs better.

Prompt Engineering. We evaluated PPTSER using diverse prompt types. Beyond the *default setting* that uses tag names as prompts, we explored an *unrelated words* setting by replacing the whole prompts with irrelevant words like `apple` and `orange`. In the *random embeddings* setting, we replaced the whole prompt's text embedding with random tensors. As Table 3 indicates, the default setting yields the highest score, suggesting the pre-trained model does grasp the prompt's semantics, and tag semantics can direct the SER task. This indicates the careful selection and design of prompts can markedly influence model performance.

Aggregation Strategies of Attention Weights. Table 3 compares how various strategies to aggregate attention weights across different heads affect performance. While *default setting* and *mean* refer to obtain the maximum and average value across attention weighs of distinct heads, *single head* uses just a single head of the attention weights to generate the final probability. The results indicate that the *max* operation outperforms others, which aligns with our hypothesis that different attention heads focus on entities with different semantics.

Parameter Efficiency. The parameter comparisons of our PPTSER methods and traditional fine-tuning are presented in Table 4. As the parameters might vary across diverse models and benchmarks, we offer a concise breakdown of the results from the methods we have tested. The results illustrate that our PPTSER has fewer parameters in comparison to traditional fine-tuning methods. For a more detailed analysis, please refer to Appendix E.

5 Related Works

SER on Visually-rich Documents. Although some early works of SER relied on heuristic algorithms (Simon et al., 1997; Schuster et al., 2013), the majority of research focused on neural network-based methods. Some of them leveraged textual

Methodology	BROS (AAAI 22)		LiLT (ACL 22)		LayoutLMv2 (ACL 21)		LayoutLMv3 (MM 22)	
	FT	**Ours**	FT	**Ours**	FT	**Ours**	FT	**Ours**
FUNSD	108.91M	**103.59M**	130.17M	**123.81M**	200.29M	**194.38M**	125.33M	**119.42M**
CORD	108.95M	**103.59M**	130.22M	**123.81M**	200.33M	**194.38M**	125.96M	**119.42M**
XFUND-zh	-	-	130.17M	**123.81M**	200.29M	**194.38M**	125.33M	**119.42M**

Table 4: Parameters of our PPTSER and traditional Fine-tuning methods. The metric in **Bold** indicates the method with fewer parameters. *FT* refers to *Fine-tuning* method.

features (Chiu and Nichols, 2016), visual features (Guo et al., 2019), or combined them with layout features (Yu et al., 2021; Wang et al., 2021a) to address this issue, but the emergence of multi-modal pre-trained models has revolutionized SER. These models are jointly pre-trained on a large-scale unlabeled document dataset with textual, layout, and even visual cues, so they have the potential to better understand a structured document. LayoutLM (Xu et al., 2020) was the first to combine textual and OCR positional features at the pre-training stage. Later, LayoutLMv2 (Xu et al., 2021a) and LayoutLMv3 (Huang et al., 2022b) further integrated visual features into the pre-training process with different architectures. Moreover, Wang et al. (2022a) advanced the model architecture with a language-agnostic layout transformer in their work, LiLT. Alongside the advancements in model structures, other works (Appalaraju et al., 2021; Li et al., 2021b,a; Hong et al., 2022; Luo et al., 2023) have focused on the diverse pre-training tasks to facilitate the fusion of diverse modalities at pre-training stage. While these advancements have improved SER capabilities to some extent, their few-shot learning abilities still require further examination.

Few-shot SER on Visually-rich Documents. Unlike SER, few-shot SER is not fully explored yet. Cheng et al. (2020) proposed to utilize graph-matching techniques (Zanfir and Sminchisescu, 2018), representing documents as graphs with nodes as OCR-scanned boxes. For an unseen document, the type of entities was determined by comparing the relationships in the graph of unseen document with those in the graphs of support documents. Yao et al. (2021) also adopted a graph-matching approach to address this issue, but the entity type was determined based on the relationships in different forms with more complex solvers. Taking a different way, Wang and Shang (2022) introduced a novel labeling scheme for SER. They reshaped SER as a generative task, and used LayoutReader (Wang et al., 2021b) for SER label generation. Although these studies preliminary explored few-shot SER, they lacked generality and pluggability, and their performances in various scenes require further exploration and improvement.

Few-shot NER in Plain Texts. While few-shot SER on visually-rich documents has seen limited exploration, there has been extensive research on few-shot Named Entity Recognition (NER) in plain texts (Wang et al., 2022b; Das et al., 2022; Ma et al., 2022a; Cheng et al., 2023). However, only few of these studies have considered the scenario where only limited data in the target domain is available. Huang et al. (2022a) proposed using the NER tag as a prompt and employing contrastive learning to address this issue. On the other hand, Ma et al. (2022b) reformulated few-shot NER as a Language Modeling task and used the pre-trained Masked Language Model head to predict a word related to the entity type for each text token. However, since these methods are designed to address the NER problem with sparse entities in plain texts, they might not be suitable for entity-rich scenarios in visually-rich documents. Additionally, they do not emphasize the issue of detecting entity boundaries, without which adjacent entities of the same type might be erroneously merged into one. The comparison with two representative few-shot NER methods also show the effectiveness of our method.

6 Conclusion

In this paper, we present PPTSER, an innovative and efficient strategy for few-shot entity recognition on visually-rich documents using a plug-and-play, tag-guided approach. PPTSER redefines the SER task as a dual-function operation of entity typing and span detection, and utilizeS the attention weight between document tokens and prompts related to SER tags as the target probability distributions. Our findings show that PPTSER is both effective and versatile in various data situations, from few-shot to full data scenarios. In the future, we plan to further investigate the capabilities of PPTSER across a range of VIE tasks like Entity Linking. In addition, we aim to explore PPTSER's potential in other few-shot scenarios, particularly those outside of the In-Label-Space setting. It is our hope that our work will spark further research and advancements in the realm of few-shot SER.

Limitations

Due to space constraints, our exploration of the few-shot SER setting is primarily limited to the In-Label-Space. Future research is essential to investigate the potential applications of our PPTSER in other few-shot settings and its adaptability to additional VIE tasks.

Ethical Considerations

Our proposed PPTSER is a purely methodological innovation, which inherently avoids direct negative social impacts. By leveraging the self-attention mechanism within multi-modal pre-trained models without adding extra modules, it does not introduce additional ethical risks beyond those already present in the existing multi-modal pre-trained models.

References

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R. Manmatha. 2021. Docformer: End-to-end transformer for document understanding. In *Proc. ICCV*, pages 993–1003.

Mengli Cheng, Minghui Qiu, Xing Shi, Jun Huang, and Wei Lin. 2020. One-shot text field labeling using attention and belief propagation for structure information extraction. In *Proc. ACM MM*, page 340–348.

Zifeng Cheng, Qingyu Zhou, Zhiwei Jiang, Xuemin Zhao, Yunbo Cao, and Qing Gu. 2023. Unifying token- and span-level supervisions for few-shot sequence labeling. *ACM Trans. Inf. Syst.*, 42(1).

Jason P.C. Chiu and Eric Nichols. 2016. Named Entity Recognition with Bidirectional LSTM-CNNs. *Transactions of the Association for Computational Linguistics*, 4:357–370.

Sarkar Snigdha Sarathi Das, Arzoo Katiyar, Rebecca Passonneau, and Rui Zhang. 2022. CONTaiNER: Few-shot named entity recognition via contrastive learning. In *Proc. ACL*, pages 6338–6353.

Jiuxiang Gu, Jason Kuen, Vlad I Morariu, Handong Zhao, Rajiv Jain, Nikolaos Barmpalios, Ani Nenkova, and Tong Sun. 2021. Unidoc: Unified pretraining framework for document understanding. In *Proc. NIPS*, volume 34, pages 39–50.

He Guo, Xiameng Qin, Jiaming Liu, Junyu Han, Jingtuo Liu, and Errui Ding. 2019. Eaten: Entity-aware attention for single shot visual text extraction. In *Proc. ICDAR*, pages 254–259.

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. *Proc. AAAI*, 36(10):10767–10775.

Yucheng Huang, Kai He, Yige Wang, Xianli Zhang, Tieliang Gong, Rui Mao, and Chen Li. 2022a. COP-NER: Contrastive learning with prompt guiding for few-shot named entity recognition. In *Proc. COLING*, pages 2515–2527.

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

Guillaume Jaume, Hazim Kemal Ekenel, and Jean-Philippe Thiran. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *Workshop at ICDAR*, volume 2, pages 1–6.

Chenliang Li, Bin Bi, Ming Yan, Wei Wang, Songfang Huang, Fei Huang, and Luo Si. 2021a. StructuralLM: Structural pre-training for form understanding. In *Proc. ACL*, pages 6309–6318.

Peizhao Li, Jiuxiang Gu, Jason Kuen, Vlad I. Morariu, Handong Zhao, Rajiv Jain, Varun Manjunatha, and Hongfu Liu. 2021b. Selfdoc: Self-supervised document representation learning. In *Proc. CVPR*, pages 5652–5660.

Yulin Li, Yuxi Qian, Yuechen Yu, Xiameng Qin, Chengquan Zhang, Yan Liu, Kun Yao, Junyu Han, Jingtuo Liu, and Errui Ding. 2021c. Structext: Structured text understanding with multi-modal transformers. In *Proc. ACM MM*, page 1912–1920.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proc. ACM SIGKDD Int. Conf. Knowledge discovery & data mining*, page 1054–1064.

Chuwei Luo, Changxu Cheng, Qi Zheng, and Cong Yao. 2023. Geolayoutlm: Geometric pre-training for visual information extraction. In *Proc. CVPR*, pages 7092–7101.

Jie Ma, Miguel Ballesteros, Srikanth Doss, Rishita Anubhai, Sunil Mallya, Yaser Al-Onaizan, and Dan Roth. 2022a. Label semantics for few shot named entity recognition. In *Findings of ACL*, pages 1956–1971.

Ruotian Ma, Xin Zhou, Tao Gui, Yiding Tan, Linyang Li, Qi Zhang, and Xuanjing Huang. 2022b. Template-free prompt tuning for few-shot NER. In *Proc. NAACL-HLT*, pages 5721–5732.

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 *Workshop on Document Intelligence at NeurIPS*.

Lance Ramshaw and Mitch Marcus. 1995. Text chunking using transformation-based learning. In *Workshop on Very Large Corpora*.

Daniel Schuster, Klemens Muthmann, Daniel Esser, Alexander Schill, Michael Berger, Christoph Weidling, Kamil Aliyev, and Andreas Hofmeier. 2013. Intellix – end-user trained information extraction for document archiving. In *Proc. ICDAR*, pages 101–105.

Anikó Simon, Jean-Christophe Pret, and A. Peter Johnson. 1997. A fast algorithm for bottom-up document layout analysis. *IEEE Trans. Pattern Anal. Mach. Intell.*, 19(3):273–277.

Erik F. Tjong Kim Sang and Fien De Meulder. 2003. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *Proc. NAACL-HLT*, pages 142–147.

Jiapeng Wang, Lianwen Jin, and Kai Ding. 2022a. LiLT: A simple yet effective language-independent layout transformer for structured document understanding. In *Proc. ACL*, pages 7747–7757.

Jiapeng Wang, Chongyu Liu, Lianwen Jin, Guozhi Tang, Jiaxin Zhang, Shuaitao Zhang, Qianying Wang, Yaqiang Wu, and Mingxiang Cai. 2021a. Towards robust visual information extraction in real world: New dataset and novel solution. *Proc. AAAI*, 35(4):2738–2745.

Rui Wang, Tong Yu, Handong Zhao, Sungchul Kim, Subrata Mitra, Ruiyi Zhang, and Ricardo Henao. 2022b. Few-shot class-incremental learning for named entity recognition. In *Proc. ACL*, pages 571–582.

Zilong Wang and Jingbo Shang. 2022. Towards few-shot entity recognition in document images: A label-aware sequence-to-sequence framework. In *Findings of ACL*, pages 4174–4186.

Zilong Wang, Yiheng Xu, Lei Cui, Jingbo Shang, and Furu Wei. 2021b. LayoutReader: Pre-training of text and layout for reading order detection. In *Proc. EMNLP*, pages 4735–4744.

Yang Xu, Yiheng Xu, Tengchao Lv, Lei Cui, Furu Wei, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, Wanxiang Che, Min Zhang, and Lidong Zhou. 2021a. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. In *Proc. ACL*, pages 2579–2591.

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 *Proc. ACM SIGKDD Int. Conf. Knowledge discovery & data mining*, page 1192–1200.

Yiheng Xu, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florêncio, Cha Zhang, and Furu Wei. 2021b. Layoutxlm: Multimodal pre-training for multilingual visually-rich document understanding. *arXiv: Comp. Res. Repository*, abs/2104.08836.

Yiheng Xu, Tengchao Lv, Lei Cui, Guoxin Wang, Yijuan Lu, Dinei Florencio, Cha Zhang, and Furu Wei. 2022. XFUND: A benchmark dataset for multilingual visually rich form understanding. In *Findings of the ACL*, pages 3214–3224.

Minghong Yao, Zhiguang Liu, Liangwei Wang, Houqiang Li, and Liansheng Zhuang. 2021. One-shot key information extraction from document with deep partial graph matching. *arXiv: Comp. Res. Repository*, abs/2109.13967.

Wenwen Yu, Ning Lu, Xianbiao Qi, Ping Gong, and Rong Xiao. 2021. Pick: Processing key information extraction from documents using improved graph learning-convolutional networks. In *Proc. ICPR*, pages 4363–4370.

Yuechen Yu, Yulin Li, Chengquan Zhang, Xiaoqiang Zhang, Zengyuan Guo, Xiameng Qin, Kun Yao, Junyu Han, Errui Ding, and Jingdong Wang. 2023. Structextv2: Masked visual-textual prediction for document image pre-training. In *Proc. ICLR*.

Andrei Zanfir and Cristian Sminchisescu. 2018. Deep learning of graph matching. In *Proc. CVPR*, pages 2684–2693.

A Few-shot Divisions Generation

To cater to the real-world application scenarios, we have organized our few-shot divisions from the full training set with Algorithm 1. Our goal was to randomly select the minimum number of documents that satisfy the *N-way K-shot* requirement of *each of the N categories has K documents containing entities of that category as the support set*.

It is worth noticing that in the context of few-shot SER on visually-rich documents, the few-shot setting of *N-way K-shot* signifies that *each of the N categories has K documents containing entities of that category as the support set*, instead of *there are K entity spans for each of N entity types as the support set* for the setting of few-shot NER on plain texts.

B Implementation Details

B.1 Implementation Details of PPTSER

We used one NVIDIA 3090 to fine-tune our model with AdamW optimizer. The learning rate is $5e-5$ with a warm up ratio of 0.1, and we fine-tuned the model for 2000 iterations with a batch size of 8 by default. Besides the default augmentation strategies for images adopted in LayoutLMv2 and LayoutLMv3, we did not employ any additional augmentation strategies.

B.2 Modification on few-shot NER method for visually-rich documents

In Section 3.3, we mentioned that we adapted two methods originally used for few-shot NER on plain text for few-shot SER on visually-rich documents. We will briefly introduce these modifications.

COPNER (Huang et al., 2022a). The COPNER method employs contrastive learning, feeding both entity label semantics and sentences into a plain text pre-trained language model. This approach uses the hidden state output of the pre-trained model to calculate a contrastive loss between sentence tokens and label semantics, then determining the entity type of tokens. However, the original COPNER could only determine if a token belonged to an entity category, without recognizing boundaries between entities. Therefore, we also improved it with the *entity typing and span detection* framework introduced in our paper. That is, while determining the entity type of tokens, we also input the tokens $beginning$ and $inner$ into the pre-trained model to detect the entity boundary. The model's output hidden state is then used to calculate a contrastive loss between sentence tokens and these $beginning$ and $inner$ tokens.

Besides, we retained this core process but replaced the original language model pre-trained on pure text with a multi-modal pre-trained model. Experiments show that our use of multi-head cross-attention methods is more suitable for SER tasks on visually-rich documents, especially in Chinese contexts.

EntLM (Ma et al., 2022b). EntLM treats NER as a task of Language Modeling. For testing a few-shot NER dataset on plain text, it first selects a related word for each entity type. Then, using the pre-trained Masked Language Modeling head of BERT, it predicts the probability distribution of each sentence token over these related words, thereby determining the probability distribution of tokens across different entity types.

The selection of related words relies on the distant data obtained from BOND (Liang et al., 2020), which uses BERT and the corpora from Wikipedia to create rough annotations for the NER test set. However, in the realm of visually-rich documents, such rough annotated data is not provided by BOND, and due to the relative abstract expression of SER tags from natural language expressions and the difference between structured documents and natural language expressions, it's not feasible to obtain rough annotations using corpora from Wikipedia with BERT. Therefore, we directly use the ground truth annotations from the SER test set as distant data to find related words associated with entity types in the SER dataset. Although the experimental results on EntLM might be artificially high due to some exposure to the entity distribution in the test set, our proposed method significantly outperforms others that only accept text modality inputs, including EntLM.

In summary, methods for few-shot NER on plain text may not necessarily transition well to the task of few-shot SER on visually-rich documents. The notable performance of our proposed method in few-shot SER on visually-rich documents further highlights the innovation and contribution of our research.

Algorithm 1: Few-shot Divisions Generation

Input: Novel Dataset with the label space $\mathbb{C} = \{c_1, c_2, ..., c_N\}$, full training set $\mathcal{D}^{full}$
Output: N-way K-shot few-shot training set $\mathcal{D}^{train}$

```
D^train = {}
Number of documents that contain entities of c_i in D^full: Q = {c_1 : 0, c_2 : 0, ..., c_N : 0}
Document set that contain entities of c_i in D^full: R = {c_1 : {}, c_2 : {}, ..., c_N : {}}
for doc_i in D^full do
    for c_j in C do
        if doc_i contain entities of c_j then
            Q[c_j] += 1
            R[c_j].append(doc_i)
        end
    end
end
Q' = sorted(Q, key = lambda x : x[1]) = {c'_1 : n_1, c'_2 : n_2, ..., c'_N : n_N} (n_1 ≤ n_2 ≤ ... ≤ n_N)
Number of documents that contain entities of c_i in D^train: S = {c_1 : 0, c_2 : 0, ..., c_N : 0}
for c'_i in keys of Q' do
    for S[c'_i] < K do
        if R[c'_i] is empty then
            break
        end
        Randomly select a document doc_candidate from R[c'_i]
        R[c'_j].pop(doc_candidate)
        if doc_candidate ∉ D^train then
            D^train.append(doc_candidate)
            for c_j ∈ C do
                if doc_candidate contain entities of c_j then
                    S[c_j] += 1
                end
            end
        else
            continue
        end
    end
end
```

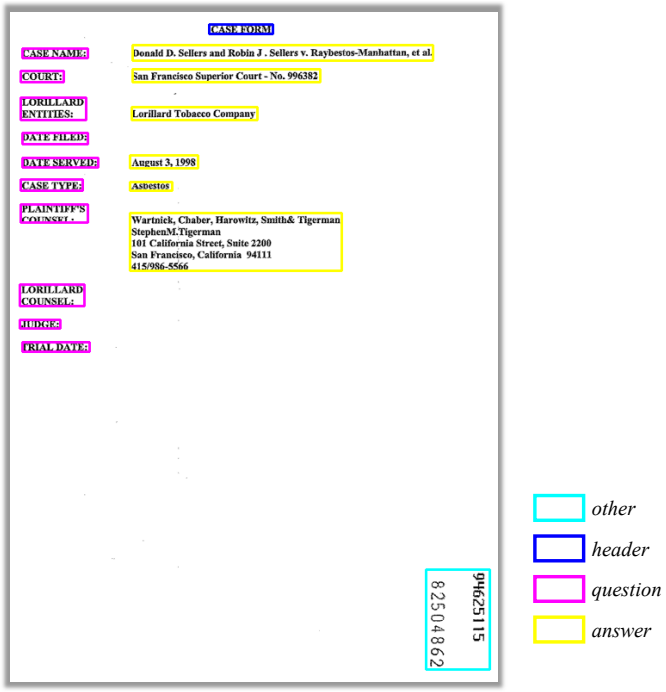

Figure 5: Illustration of a sample from FUNSD dataset. Different colored boxes represent the entities of distinct kinds. Zoom in for better view.

C Examples when apply PPTSER

To more intuitively demonstrate our method, we provide an example from FUNSD dataset when apply our PPTSER method.

The FUNSD dataset includes three meaningful entity types: *header*, *question*, and *answer*, with all other uninteresting entities categorized as *other*. Hence, the entity type set is $\mathbb{E} = \{other, header, question, answer\}$. The label spaces for entity typing and span detection are $\mathbb{C}^{ent.} = \{other, header, question, answer\}$ and $\mathbb{C}^{det.} = \{beginning, inner\}$, respectively. We directly use the label's names as tag-related prompts, with the prompts for entity typing and span detection being $\tilde{\mathbb{C}}^{ent.}$ = *"other header question answer"* and $\tilde{\mathbb{C}}^{det.}$ = *"beginning inner"*. These prompts are then concatenated to form the full tag-related prompt $\tilde{\mathbb{C}}$ = *"other header question answer beginning inner"*.

Consider an example from the FUNSD dataset shown in Figure 5, with the document content *"... CASE TYPE: Asbestos ... 82504862"*, where *"..."* indicates omitted parts. Here, *"CASE TYPE:"* belongs to the entity type of *question*, *"Asbestos"* to the entity type of *answer*, and *"82504862"* to *other*. Assuming the tokenizer splits the document into *"CASE"*, *"TYPE:"*, *"Asbestos"*, and *"82504862"*, their labels for entity typing and span detection would be $\boldsymbol{y}^{ent.} = [question, question, answer, other]$ and $\boldsymbol{y}^{det.} = [beginning, inner, beginning, -1]$.

Subsequently, *"CASE"*, *"TYPE:"*, *"Asbestos"*, and *"82504862"* as document tokens are concatenated with the full tag-related prompt, and form the boosted input $\boldsymbol{x}'$ = *"CASE TYPE: Asbestos 82504862 other header question answer beginning inner"*. Then, $\boldsymbol{x}'$ is input into the multi-modal pretrained model as the *Text Embedding* to obtain the multi-head attention weight and the aggregated attention weight from the last block, as shown in Figure 6.

During training stage, as shown in Figure 7(a), the aggregated attention weight is split into attention weights between *"other header question answer"* and document tokens, as well as *"beginning inner"* and document tokens, which are then used to calculate the losses for entity typing and span detection, respectively, culminating in a combined total loss.

During inference stage, as shown in Figure 7(b), we select the document tokens with the highest probability for *"other header question answer"* and *"beginning inner"* as $\hat{\boldsymbol{y}}^{ent.}$ and $\hat{\boldsymbol{y}}^{det.}$, then combine them to get the predictions under BIO tagging scheme $\hat{\boldsymbol{y}}$ following the procedure in Section 2.2.3. And the entity spans are finally analyzed from $\hat{\boldsymbol{y}}$.

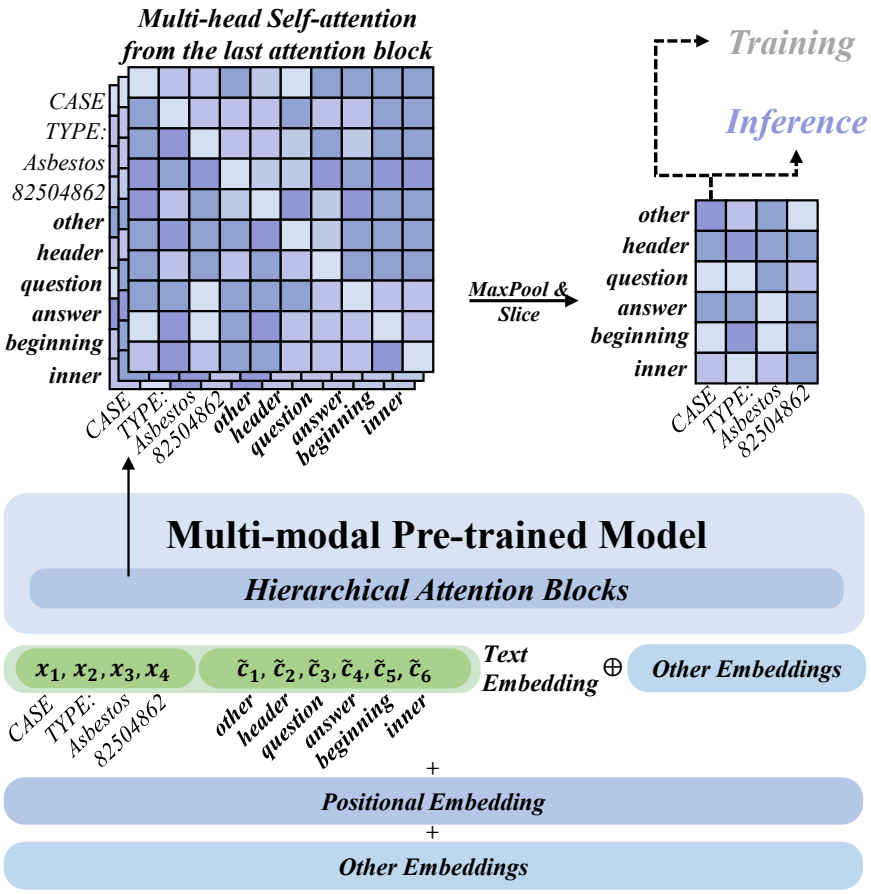

Figure 6: Example of a FUNSD sample run with PPTSER. In the image, **bolded words** in ***Text Embeddings*** indicate they are part of the prompt. In the Multi-head Self-attention section, the brightness of the color represents the magnitude of the value.

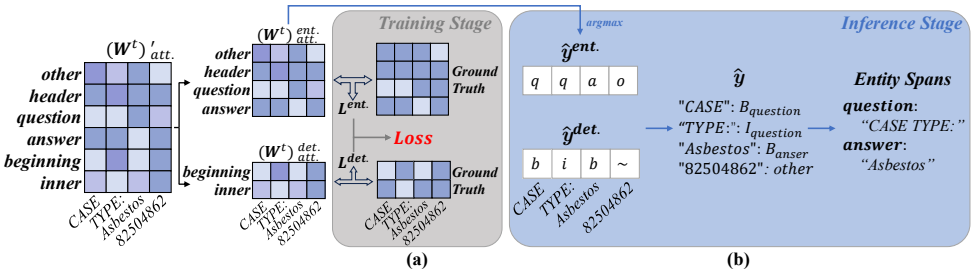

Figure 7: Example of a FUNSD sample running on PPTSER at *Training stage* and *Inference stage*. In the *Inference stage*, q, a, b, i and o respectively stand for the *question*, *answer*, *beginning*, *inner* and *other* categories.

Modality		Text + Layout				Text + Layout + Image			
Methodology		BROS (AAAI 22)		LiLT (ACL 22)		LayoutLMv2 (ACL 21)		LayoutLMv3 (MM 22)	
		FT	**Ours**	FT	**Ours**	FT	**Ours**	FT	**Ours**
FUNSD	1-shot	49.17	**52.50**	**51.07**	50.91	44.15	**46.91**	42.67	**56.27**
	3-shot	63.07	**64.31**	65.65	**67.24**	**60.67**	60.65	72.66	**75.18**
	5-shot	66.31	**68.10**	71.11	**72.95**	63.45	**64.36**	77.29	**79.53**
	7-shot	69.24	**71.06**	72.49	**74.14**	65.94	**66.79**	79.22	**81.31**
	Full Data	83.42	**83.67**	88.62	**88.89**	83.54	**83.59**	91.41	**91.96**
CORD	1-shot	64.92	**68.04**	69.31	**75.35**	68.08	**69.73**	70.03	**74.02**
	3-shot	78.75	**79.26**	81.63	**83.85**	79.92	**81.35**	81.97	**85.09**
	5-shot	83.86	**84.23**	85.62	**87.03**	83.84	**84.42**	85.76	**87.71**
	7-shot	83.56	**83.76**	85.39	**87.74**	84.40	**85.09**	86.92	**88.37**
	Full Data	95.72	**95.88**	95.82	**96.06**	94.95	**95.64**	96.34	**96.39**
XFUND-zh	1-shot	-	-	59.03	**64.90**	59.17	**67.04**	46.64	**54.59**
	3-shot	-	-	71.33	**71.54**	73.17	**75.21**	62.73	**72.36**
	5-shot	-	-	75.09	**77.07**	79.21	**79.96**	69.39	**76.57**
	7-shot	-	-	77.34	**77.70**	79.68	**80.22**	72.03	**77.95**
	Full Data	-	-	87.92	**88.17**	88.60	**88.95**	89.09	**91.04**

(a) Precision (%) of our PPTSER and Traditional Fine-tuning methods.

Modality		Text + Layout				Text + Layout + Image			
Methodology		BROS (AAAI 22)		LiLT (ACL 22)		LayoutLMv2 (ACL 21)		LayoutLMv3 (MM 22)	
		FT	**Ours**	FT	**Ours**	FT	**Ours**	FT	**Ours**
FUNSD	1-shot	49.75	**57.89**	54.39	**61.77**	53.64	**59.15**	53.14	**70.07**
	3-shot	65.82	**71.50**	69.81	**71.35**	62.82	**67.01**	76.94	**78.64**
	5-shot	69.32	**73.43**	75.61	**77.76**	68.50	**70.97**	81.92	**83.65**
	7-shot	67.54	**72.94**	74.33	**77.38**	67.23	**71.04**	80.51	**81.91**
	Full Data	**84.26**	84.15	**89.30**	89.26	83.51	**83.85**	90.91	**92.05**
CORD	1-shot	67.70	**68.95**	70.79	**75.80**	**71.23**	70.22	70.67	**74.35**
	3-shot	79.30	**79.96**	81.65	**83.82**	81.36	**81.98**	82.14	**85.45**
	5-shot	84.21	**84.52**	85.43	**87.08**	**84.81**	84.63	85.91	**87.84**
	7-shot	83.79	**84.43**	85.32	**87.72**	85.12	**85.54**	86.97	**88.59**
	Full Data	**95.72**	95.61	95.78	**96.03**	95.45	**95.62**	96.34	**96.40**
XFUND-zh	1-shot	-	-	61.54	**70.68**	61.46	**69.68**	**61.21**	59.24
	3-shot	-	-	74.22	**77.22**	75.88	**79.46**	77.24	**78.58**
	5-shot	-	-	79.99	**82.02**	83.88	**85.02**	**82.27**	82.23
	7-shot	-	-	83.97	**85.44**	85.03	**87.43**	**84.76**	84.30
	Full Data	-	-	93.18	**93.20**	91.96	**92.72**	**94.27**	93.37

(b) Recall (%) of our PPTSER and Traditional Fine-tuning methods.

Table 5: Precision and Recall of PPTSER and Traditional Fine-tuning methods. Metrics in **Bold** is better between PPTSER and Fine-tuning and *FT* refers to *Fine-tuning* methods.

D Further Analysis of Experimental Results

This section presents further analysis and additional performance metrics obtained from the main experiments.

D.1 Further Analysis of Comparisons with Existing Fine-Tuning Methods

Table 5a and Table 5b present the precision and recall of PPTSER compared to the traditional fine-tuning method. The results demonstrate that PPTSER consistently outperforms the traditional fine-tuning method in most cases, leading to improved overall performance in terms of F1 scores.

Additionally, we observed from Table 1 and Table 5 that the improvement of PPTSER on the CORD dataset is generally less pronounced compared to its performance on other benchmarks. This prompted further investigation on our part.

As mentioned in Section 2.1, our *N-way K-shot* setting implies that *each of the N categories has K documents containing entities of that category as the support set*. Given that a single document in the CORD dataset cannot encompass entities from all categories, we selected more than K documents under the K-shot setup in the previous experiments. Consequently, we reselected 1 to 5 document samples as the training set for CORD and conducted additional experiments with LiLT, which showed the

Methodology		LayoutLMv2		LiLT	
		FT	**Ours**	FT	**Ours**
Sample Number	1	33.28	**33.81**	40.46	**42.47**
	2	44.86	**47.09**	49.39	**54.18**
	3	54.24	**56.91**	56.31	**63.71**
	4	61.54	**62.23**	61.36	**69.19**
	5	63.89	**64.92**	64.09	**69.83**

(a) F1 score (%) of our PPTSER and Traditional Fine-tuning methods on CORD benchmark.

Methodology		LayoutLMv2		LiLT	
		FT	**Ours**	FT	**Ours**
Sample Number	1	33.10	**34.06**	40.35	**40.08**
	2	42.31	**46.20**	47.26	**52.58**
	3	51.31	**57.50**	53.58	**62.92**
	4	59.58	**62.50**	60.17	**68.72**
	5	61.93	**64.65**	63.21	**69.47**

(b) Precision (%) of our PPTSER and Traditional Fine-tuning methods on CORD benchmark.

Methodology		LayoutLMv2		LiLT	
		FT	**Ours**	FT	**Ours**
Sample Number	1	33.82	**33.75**	41.36	**45.28**
	2	48.01	**48.10**	51.81	**55.89**
	3	**57.60**	56.35	57.73	**64.52**
	4	63.64	**61.97**	62.63	**69.68**
	5	66.00	**65.22**	65.01	**70.19**

(c) Recall (%) of our PPTSER and Traditional Fine-tuning methods on CORD benchmark.

Table 6: Performances of our PPTSER and Traditional Fine-tuning methods on CORD benchmark with various numbers of sample as the support set. Metrics in **Bold** is the best and *FT* refers to *Fine-tuning* methods.

most significant improvement, and LayoutLMv2, which showed the least. The results, as illustrated in Table 6, led to several conclusions:

- **Sample Size**: The K-shot division on CORD often includes more than K samples. For example, in the 1-shot experiment, our training set averaged 7.6 samples. In contrast, on FUNSD, a single sample typically encompasses entities of all 4 types, resulting in only 1 sample in the training set for the 1-shot scenario. Our findings indicate that as the sample size increases, the performance gap between our method and conventional fine-tuning diminishes, yet our method retains its advantage. Therefore, under the same K-shot setting, CORD involves more samples than other datasets, which leads to a less pronounced improvement.

- **Label Complexity**: We believe that the labels in CORD are more complex and abstract, making it harder for the model to grasp their semantic meanings compared to those in FUNSD. For instance, entity categories in FUNSD include *header*, *question*, and *answer*, whereas in CORD, they involve more abstract types like *menu.num* and *total.creditcardprice*. The experimental results indicate that when the sample size is extremely small, the improvement on CORD from our method is limited. However, this improvement increases rapidly with the sample size, suggesting our method can more accurately capture the relationship between document tokens and tag-related prompts with relatively more samples.

- **Pre-trained Models**: Different pre-trained models have varying degrees of understanding of labels. This could explain why some pre-trained models show weaker improvement on CORD. Our supplementary experiments reveal that the extent of improvement of our method is consistently better with the LiLT than the LayoutLMv2, suggesting that LiLT better understands the semantic information implied by the labels.

D.2 Further Analysis of Comparisons with Existing Few-shot Methods

We also present a detailed comparison of PPTSER with other few-shot methods, including precision and recall metrics in Table 7a and Table 7b. Similar to the F1 score, models enhanced with PPTSER usually demonstrate superior performance compared to both few-shot NER and few-shot SER methods.

Modality		Text		Text + Layout			Text + Layout + Image	
Methodology		EntLM (NAACL 22)	COPNER (COLING 22)	LASER (ACL 22)	COPNER$_{LiLT}$ (COLING 22)	**PPTSER$_{LiLT}$ (Ours)**	COPNER$_{LMv3}$ (COLING 22)	**PPTSER$_{LMv3}$ (Ours)**
FUNSD	1-shot	22.85	18.67	36.61	53.79	50.91	49.01	**56.27**
	3-shot	33.39	30.97	46.71	67.26	67.24	73.54	**75.18**
	5-shot	37.23	32.57	46.80	71.40	72.95	75.40	**79.53**
	7-shot	40.29	35.43	51.17	72.81	74.14	78.63	**81.31**
	Full Data	67.53	63.39	69.08	87.45	88.89	91.45	**91.96**
CORD-Lv1	1-shot	73.23	67.42	65.56	86.93	**90.62**	86.80	90.05
	3-shot	82.85	81.37	75.43	94.39	94.97	93.99	**95.14**
	5-shot	86.87	85.87	82.07	94.94	**96.38**	95.65	96.20
	7-shot	86.65	86.54	83.54	95.12	96.25	96.05	**96.53**
	Full Data	95.93	95.83	96.50	99.23	99.43	**99.45**	**99.45**
CORD	1-shot	57.45	51.22	-	70.27	**75.35**	67.14	74.02
	3-shot	71.42	67.26	-	81.48	83.85	79.89	**85.09**
	5-shot	77.70	74.59	-	84.92	87.03	85.13	**87.71**
	7-shot	78.51	75.69	-	85.86	87.74	86.83	**88.37**
	Full Data	93.56	92.33	-	95.75	96.06	95.79	**96.39**
XFUND-zh	1-shot	27.02	23.98	-	49.49	**64.90**	53.37	54.59
	3-shot	35.94	35.72	-	64.69	71.54	69.98	**72.36**
	5-shot	43.18	42.93	-	68.21	**77.07**	73.48	76.57
	7-shot	45.13	44.66	-	72.61	77.70	76.42	**77.95**
	Full Data	64.75	65.59	-	87.23	88.17	**91.10**	91.04

(a) Precision (%) of PPTSER and other Few-shot methods.

Modality		Text		Text + Layout			Text + Layout + Image	
Methodology		EntLM (NAACL 22)	COPNER (COLING 22)	LASER (ACL 22)	COPNER$_{LiLT}$ (COLING 22)	**PPTSER$_{LiLT}$ (Ours)**	COPNER$_{LMv3}$ (COLING 22)	**PPTSER$_{LMv3}$ (Ours)**
FUNSD	1-shot	28.01	21.53	41.05	56.72	61.77	54.43	**70.07**
	3-shot	37.56	32.12	46.55	70.17	71.35	78.32	**78.64**
	5-shot	42.42	38.28	52.14	75.59	77.76	79.83	**83.65**
	7-shot	42.79	39.57	54.08	73.92	77.38	78.45	**81.91**
	Full Data	67.31	65.81	69.41	88.04	89.26	91.06	**92.05**
CORD-Lv1	1-shot	75.38	69.90	68.12	87.01	**90.38**	87.16	89.99
	3-shot	84.54	83.17	76.77	93.95	94.61	94.06	**95.12**
	5-shot	87.35	86.30	82.42	94.78	96.04	95.84	**96.23**
	7-shot	87.99	86.97	83.71	94.96	96.00	96.07	**96.50**
	Full Data	95.94	95.97	96.62	99.19	99.42	**99.45**	**99.45**
CORD	1-shot	58.29	53.79	-	69.83	**75.80**	67.51	74.35
	3-shot	71.95	68.35	-	81.08	83.82	80.24	**85.45**
	5-shot	77.78	74.69	-	84.68	87.08	85.47	**87.84**
	7-shot	78.76	76.51	-	85.66	87.72	86.91	**88.59**
	Full Data	93.45	92.68	-	95.72	96.03	95.79	**96.40**
XFUND-zh	1-shot	26.35	23.26	-	48.26	**70.68**	55.67	59.24
	3-shot	39.62	40.48	-	64.88	77.22	72.89	**78.58**
	5-shot	44.03	46.14	-	70.00	82.02	79.92	**82.23**
	7-shot	48.46	49.58	-	76.58	**85.44**	82.47	84.30
	Full Data	67.72	68.71	-	91.20	93.20	92.91	**93.37**

(b) Recall (%) of PPTSER and other Few-shot methods.

Table 7: Precision and Recall of PPTSER and other Few-shot methods. Metrics in **Bold** is the best, and that with underline is the second best.

	CORD					
	7th *block*	8th *block*	9th *block*	10th *block*	11th *block*	12th *block*
1-shot	68.27	69.63	68.34	68.93	74.27	75.57
3-shot	80.09	81.16	80.81	81.51	82.89	83.83
5-shot	84.29	85.25	85.00	84.84	86.41	87.06
7-shot	84.58	85.86	85.48	85.97	86.79	87.73
Full Data	95.63	95.59	95.48	96.12	95.86	96.04
Δ	**-3.47**	**-2.55**	**-3.03**	**-2.57**	**-0.80**	-

(a) F1 score (%) of PPTSER on CORD benchmark when obtaining the attention weight from different blocks.

	CORD					
	7th *block*	8th *block*	9th *block*	10th *block*	11th *block*	12th *block*
1-shot	67.36	69.02	67.86	68.53	73.91	75.35
3-shot	79.76	81.19	80.78	81.46	82.76	83.85
5-shot	80.98	85.38	85.01	84.78	86.22	87.03
7-shot	84.41	85.92	85.52	85.97	86.68	87.74
Full Data	95.70	95.64	95.55	96.16	95.87	96.06
Δ	**-4.36**	**-2.58**	**-3.06**	**-2.63**	**-0.92**	-

(b) Precision (%) of PPTSER on CORD benchmark when obtaining the attention weight from different blocks.

	CORD					
	7th *block*	8th *block*	9th *block*	10th *block*	11th *block*	12th *block*
1-shot	69.21	70.25	68.83	69.34	74.64	75.80
3-shot	80.43	81.14	80.84	81.56	83.03	83.82
5-shot	80.41	85.13	84.99	84.89	86.60	87.08
7-shot	84.75	85.80	85.43	85.97	86.90	87.72
Full Data	95.57	95.54	95.41	96.09	95.85	96.03
Δ	**-4.02**	**-2.52**	**-2.99**	**-2.52**	**-0.69**	-

(c) Recall (%) of PPTSER on CORD benchmark when obtaining the attention weight from different blocks.

Table 8: Performances of PPTSER on CORD benchmark when obtaining the attention weight from different blocks. Δ denotes the average deviation of the corresponding metrics compared to that in the 12th *block*.

D.3 Further Analysis of Attention weights obtained from different blocks

We additionally provide the numerical metrics of distinct settings to obtain the attention weight from different blocks. Table 8 illustrates the experimental results, indicating that obtaining attention weights from the last block yields the best performance of F1 score, precision, and recall. Although the reduction of parameters alleviates over-fitting to some extent, since some shallower blocks outperform certain deeper ones in the 1-shot scenario, our default setting to obtain the attention weight from the last block significantly outperforms the alternative settings of obtaining the attention weight from shallower blocks. This finding strongly reinforces the effectiveness of our design.

D.4 Further Analysis of Different Designs on PPTSER

Besides, we also offer the precision and recall of our PPTSER under different designs in Table 9a and Table 9b. The outcomes clearly indicate that our PPTSER, characterized by its meticulous design, outperforms other designs across all metrics evaluated, including the F1 score, Precision, and Recall. This superiority not only showcases the robustness of our design but also significantly substantiates the efficacy of our PPTSER framework in tackling the few-shot SER problem on visually-rich documents.

Model Designs	**Baseline** *default setting*	**Decoupling Strategies** *plain BIO prompt*	**Prompt Engineering** *unrelated words*	*random embeddings*	**Aggregation Stratergies** *mean*	*single head*
1-shot	75.35	73.92	70.58	71.32	74.16	74.92
3-shot	83.85	82.44	82.83	83.24	83.04	84.07
5-shot	87.03	86.50	85.84	86.47	86.61	86.97
7-shot	87.74	86.64	86.44	87.09	86.76	87.30
Full Data	96.06	95.58	96.27	95.75	96.23	96.08
Δ	-	**-0.99**	**-1.61**	**-1.23**	**-0.64**	**-0.14**

(a) Precision (%) of PPTSER on CORD benchmark with different designs.

Model Designs	**Baseline** *default setting*	**Decoupling Strategies** *plain BIO prompt*	**Prompt Engineering** *unrelated words*	*random embeddings*	**Aggregation Stratergies** *mean*	*single head*
1-shot	75.80	74.66	71.03	71.54	74.69	75.51
3-shot	83.82	82.37	82.54	82.90	83.06	83.77
5-shot	87.08	86.01	85.72	86.11	86.67	86.99
7-shot	87.72	86.35	86.40	86.81	86.71	87.19
Full Data	96.03	95.00	96.24	95.70	96.18	96.03
Δ	-	**-1.21**	**-1.70**	**-1.48**	**-0.63**	**-0.19**

(b) Recall (%) of PPTSER on CORD benchmark with different designs.

Table 9: Precision and Recall of PPTSER on CORD benchmark with different designs. Δ denotes the average deviation of the corresponding metrics relative to the *default setting*.

E In-depth Parameter Analysis of PPTSER over Traditional Fine-tuning

We provide a further analysis of the parameter counts in this section. As shown in Table 4, our PPTSER maintains consistent parameters across different benchmarks with the same pre-trained model. This is attributed to the fact that PPTSER does not necessitate an additional classifier layer, unlike the traditional fine-tuning method. Consequently, the parameter variance arises when employing the traditional fine-tuning method with the same pre-trained models on different benchmarks, owing to variations in the number of entity types present in those benchmarks. Furthermore, as PPTSER omits the value transform layer and the feed-forward layer in the final attention block, we achieve a reduction in the parameter count of the pre-trained model it is based on. Additionally, the extent of parameter reduction varies among different pre-trained models due to disparities in their architectural designs, resulting in slice differences in the eliminations of the modules.