Towards Few-shot Entity Recognition in Document Images: A Graph Neural Network Approach Robust to Image Manipulation

Prashant Krishnan, Zilong Wang, Yangkun Wang, Jingbo Shang
University of California, San Diego
pvaidyanathan@ucsd.edu, zlwang@ucsd.edu, yaw048@ucsd.edu, jshang@ucsd.edu

Abstract

Recent advances of incorporating layout information, typically bounding box coordinates, into pre-trained language models have achieved significant performance in entity recognition from document images. Using coordinates can easily model the position of each token, but they are sensitive to manipulations in document images (e.g., shifting, rotation or scaling) which are common in real scenarios. Such limitation becomes even worse when the training data is limited in few-shot settings. In this paper, we propose a novel framework, LAGER, which leverages the topological adjacency relationship among the tokens through learning their relative layout information with graph neural networks. Specifically, we consider the tokens in the documents as nodes and formulate the edges based on the topological heuristics. Such adjacency graphs are invariant to affine transformations, making it robust to the common image manipulations. We incorporate these graphs into the pre-trained language model by adding graph neural network layers on top of the language model embeddings. Extensive experiments on two benchmark datasets show that LAGER significantly outperforms strong baselines under different few-shot settings and also demonstrate better robustness to manipulations.

Keywords: Entity Recognition, Document Image Understanding, Graph Neural Networks, BERT

1. Introduction

Entity recognition is a fundamental task in document image understanding which aims at identifying and extracting specific segments of text in the document which serve as *header*, *question* or *answer*. However, the named entity recognition in document images is different from the traditional text-only counterparts since document images, such as tables, receipts and forms, involves richer information though the layout structure. The complex layout and format of these document images provide additional information that can be used to enhance the performance of entity recognition beyond what is possible with only text. Therefore, they present an ideal scenario to use multi-modal techniques.

Recent existing methods use large self-supervised pre-trained models (Xu et al., 2020, 2021; Huang et al., 2022) for named-entity recognition in document images. These approaches extract the word spans using the standard `IOBES` tagging schemes (Marquez et al., 2005; Ratinov and Roth, 2009) in named entity recognition tasks. The models inherit the architecture from the text-only language models, such as BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), extend the embedding layer with the layout information, and build layout-aware attention mechanisms. These approaches typically leverage the bounding box coordinates to capture the overall structure of the document, which is straight-forward and has proven to be effective. However, we argue that these coordinates-based approaches fail to properly cope with image manipulation, such as shifting, rotation and scaling, which is common in real life. These image manipulations make it challenging for coordinate-based approaches to accurately understand the documents, as the coordinates can be significantly altered and the spatial relationships learned by these coordinates are no longer valid.

Given the aforementioned challenges, we propose LAGER, a layout-aware graph-based entity recognition model. Our new framework further exploits the structural information in these document images utilizing the topological relationship of the entities. We make use of graph neural networks to encode topological relationship in the document. Such practice has been proven effective in other domains such as the web mining from semi-structured web pages (Lockard et al., 2019, 2020), where they build rich representations for text fields on a web page with graphs. We construct graphs based on the spatial relationship in the document images where the entities correspond to the nodes and the edges are constructed according to heuristics relating to distance and angles between them. In this way, the topological relationship of the entities are explicitly encoded and the resulting graph is robust to the image manipulations mentioned above. We use a Graph Attention Network (GAT) (Veličković et al., 2018) to encode the graph in the latent space and combine it with the rich representations from the pre-trained language models. LAGER serves as an additional component for the existing layout-aware language models to enhance their robustness to image manipulations and extend their capacity to handle document images with complex

layouts. Our approach is particularly useful in few-shot settings when there is limited data availability for entity recognition, as the graph-based method is efficient to train and easier to generalize.

As shown in Figure 1, LAGER extends the architecture of a layout-aware language model which we use as a backbone for our framework (Xu et al., 2021; Huang et al., 2022). We construct graphs where the nodes correspond to the words in the documents. The edges are constructed based on either k-nearest neighbors of the bounding boxes in space or at multiple angles (the detailed description is given in Section 3.3). The adjacency matrix of this graph along with the hidden states of the backbone language model are given as inputs to the graph attention network (GAT). The enhanced output embeddings from the GAT are then used to perform classification.

We validate our model using two benchmarks, FUNSD (Guillaume Jaume, 2019) and CORD (Park et al., 2019). Both datasets are from real scenarios and fully-annotated with textual contents and bounding boxes. We compare our model with strong baselines and also show how our model is robust to image alterations such as rotations, scaling and shifting. We summarize our contribution as follows.

- We propose a novel framework LAGER that improves existing language models by utilizing the topological relationship of the entities in the document images with Graph Attention Networks.
- We show that our approach is robust to image manipulations such as scaling, shifting or rotating, and it is effective to various layout-aware language models.
- Extensive experiments on two benchmark datasets and two backbone models demonstrate the effectiveness of LAGER under few-shot settings.

Reproducibility. The code and the datasets will be released on Github[1].

2. Related Work

Layout-aware LMs. Given that post-OCR processing has huge potential for various downstream tasks, there are many existing works that have adapted the pre-training in language models such as BERT (Devlin et al., 2018) to include layout-information. LayoutLM (Xu et al., 2020) was the first to successfully incorporate layout information in the form of coordinates into the embedding layer of BERT. Following LayoutLM, there was LayoutLMv2 (Xu et al., 2021) which leveraged visual features and improved alignment between words and regions on the page. LayoutLMv3 (Huang et al., 2022) like LayoutLMv2 did use visual features but unified text and image masking objectives. There have been other multimodal transformer models such as DocFormer (Appalaraju et al., 2021) which uses text, vision and spatial features. They combine these features using a novel multi-modal self-attention layer. MGDoc (Wang et al., 2022) aims to exploit the spatial hierarchical relationships between content at different levels of granularity in document images. They do this by encoding page-level, region level, and word-level information at the same time into the pre-training framework.

Few-shot methods. Recently in the Visually-rich Document Understanding (VrDU) domain, there have been efforts to build robust models under few-shot settings. For semi-structured documents such as business documents, a domain agnostic few-shot learning approach was used (Mandivarapu et al., 2021). Using deep canonical correlation, they were able align the extracted text and image feature vectors. More recently for entity recognition in document images, LASER (Wang and Shang, 2022) used a label-aware seq2seq framework. They followed a new labeling scheme that generates the label surface names word-by-word explicitly after generating the entities in few-shot settings.

Graphs in multimodal few-shot settings. Graphs are an extremely useful and general representation of data. This is especially true, when there is some relationship between the data objects in question. Opencerces (Lockard et al., 2019) for the task of open information extraction from semi structured websites, utilized graphs for their semi-supervised learning approach. ZeroShotCeres (Lockard et al., 2020), as a successor to OpenCeres used a graph neural network-based approach to build rich representations of text fields on a webpage. They built graphs where each text field became nodes in graphs and used the relationships between the text fields to connect the edges. More recently, FormNet and FormNetv2 (Lee et al., 2022, 2023) used graph convolutions to aggregate semantically meaningful information in tokens present in document images.

3. Methodology

3.1. Task Formulation

Few-shot entity recognition in document images is a subtask of information extraction that seeks to locate and classify named entities into categories using a limited number of training examples. A document image $\mathcal{P}$, consists of textual and layout information. The textual contents correspond to the words, w, in the document image and we also have their annotated bounding boxes denoted by $B = (x_0, y_0, x_1, y_1)$ (where (x_0, y_0) and (x_1, y_1) are

[1] github.com/prash29/LAGER

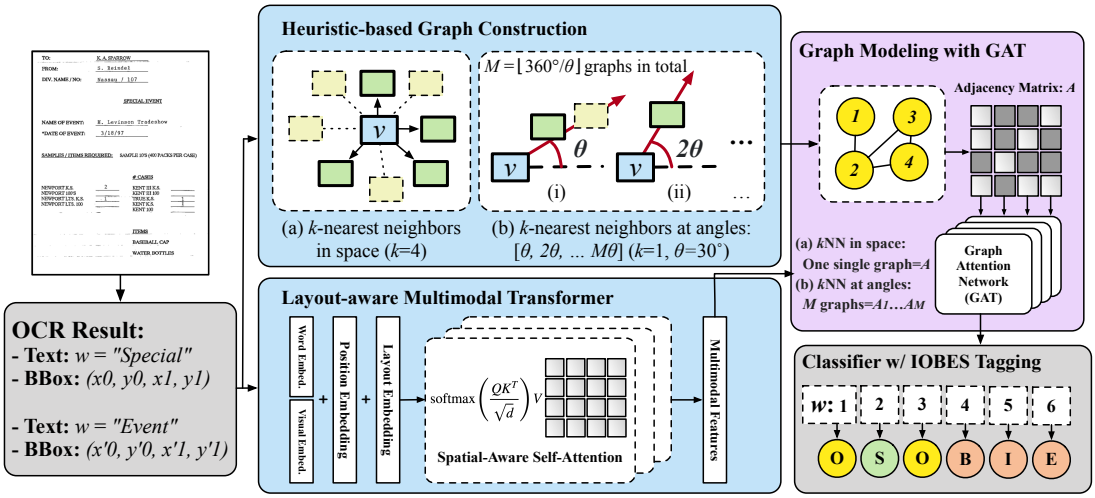


Figure 1: The Framework for LAGER. Two variants of the model are used based on the heuristic for graph construction. M denotes the number of GATs used. $M = 1$ for k-nearest neighbors in space approach. For the the k-nearest neighbors at angles, $M = \lfloor 360°/\theta \rfloor$ and we construct graphs for $[\theta, 2\theta ... M\theta]$.

the top-left and bottom-right corners). These annotations are done by human annotators or OCR engines. These words and bounding boxes are listed sequentially and act as inputs for the textual and layout modalities. The entities are defined as spans of words referring to specific concepts in the document. For example, in FUNSD, the entities correspond to `question`, `answer` or `header`. We train the model with a small subset of training samples (few shots), and test it with full testing set. We denote the the number of training samples as f in f-shot training.

3.2. Pre-trained LM as Backbone

We perform a thorough literature review and found open source models used in this domain in works such as Wang and Shang (2022). We find BERT(Devlin et al., 2018), RoBERTa(Liu et al., 2019), LayoutLM(Xu et al., 2020), LayoutLMv2(Xu et al., 2021) and LayoutLMv3(Huang et al., 2022) as models representative for this task. From Table 6 in the Appendix, we pick the strongest two baselines among these, i.e. LayoutLMv2 and LayoutLMv3. LAGER is built upon layout aware pre-trained language models such as LayoutLMv2 (Xu et al., 2021) or LayoutLMv3 (Huang et al., 2022). These models are multi-modal transformer models which take text, visual, and layout information as input to incorporate the different interactions. The layout information used are the bounding box coordinates of the tokens in the document. The output hidden states from the language model is denoted in the form of a feature matrix, $H \in \mathcal{R}^{N \times d}$, where d represents the dimension of the hidden state and N denotes the number of tokens in the document.

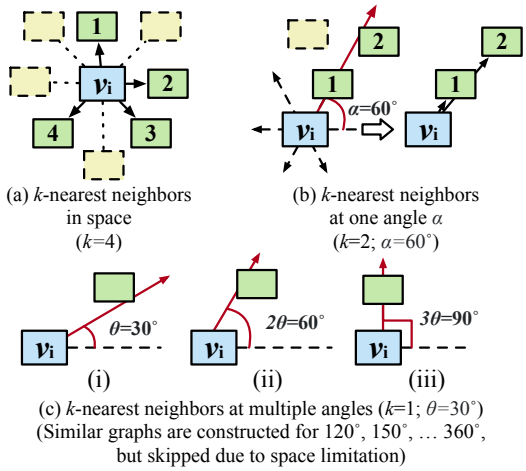


Figure 2: Heuristics for graph construction.

3.3. Heuristic-based Graph Construction

As discussed previously, the idea of constructing graphs for our document images is to exploit the topological or adjacency relationship present in the entities in the document. Towards this, we construct graphs based on certain heuristics that are used as inputs to the Graph Attention Network (GAT, described in Section 3.4).

Node and Edge definition. Given a document image page $\mathcal{P}$ with N tokens denoted by $T = \{t_1, t_2, ..., t_N\}$, let t_i refer to the i-th token in a text sequence in the dataset. For the token t_i, we also know the coordinates of its bounding box, $B_i = (x_{i0}, y_{i0}, x_{i1}, y_{i1})$. Thus for our graph $G = (V, E)$, the vertices $V = \{v_1, v_2, ..., v_N\}$ correspond to all the tokens T and their corresponding bounding boxes. The edges E represent the relationship between pairs of vertices or tokens.

We construct an undirected graph, where an edge $e_{ij} \in E$ connects two vertices v_i and v_j. Now, we describe how these edges are constructed.

Graph construction. We build graphs based on two heuristics which primarily relate to the Euclidean distance between two tokens in a document. The edges in the graph between tokens are constructed based on either of the following heuristics:

- **k-nearest neighbors in space:** For a token t_i, we calculate the Euclidean distance between the corresponding token with all other tokens in T. We form edges between t_i and its k-closest tokens. If t_i is a k-nearest neighbor of t_j, there is an edge, or if t_j is a k-nearest neighbor of t_i, there is an edge. A representative example for $k = 4$ is shown in Figure 2.a.

- **k-nearest neighbors at multiple angles:** We first describe our method to find the k-nearest neighbor at one angle α. Basically, for a token t_i, the edges formed are restricted to the k-nearest tokens in the direction of α. An example for $k = 2$ and $\alpha = 60°$ is shown in Figure 2.b.
  - We draw a ray from the centroid of a token's bounding box that forms an angle (α) with the x-axis (the red ray in Figure 2.b).
  - We find all tokens such that the ray intersects with any part of the token's bounding box. We then select the k-nearest such tokens (the token 1 and 2 in Figure 2.b).

  The approach for one single angle collects the information in that particular direction. We create multiple graphs to represent the global topological relation of each token. We pick an angle θ and $M = \lfloor 360°/\theta \rfloor$ graphs are created with $\alpha \in \{\theta, 2\theta, ...\}$ (the graphs in Figure 2.c).

After constructing the graph(s), we would create one adjacency matrix A for k-nearest neighbors in space or multiple adjacency matrices $A_1, ..., A_M$ for k-nearest neighbors at multiple angles. For simplicity, we denote these adjacency matrices by $A \in \mathcal{R}^{N \times N}$ to represent the topological structure when there is no ambiguity. And $A_{v_i, v_j} = 1$ if and only if an edge $e = (v_i, v_j)$ exists in our graph.

We believe that constructing the graph using the heuristics described above allows us to capture some relationships between tokens in the document that is not leveraged by using just the layout aware pre-trained language model. The graphs, especially the one constructed using k-nearest neighbors at multiple angles can preserve the topological relationship. It also helps in recovering the relative positions of the different bounding boxes even in cases described in Section 3.5 where the documents are altered with scaling, rotations or shifting.

3.4. Graph Modeling with GAT

Our model combines the representations from the pre-trained language model with the graph we construct for a document described in the previous section. For this, we use a graph neural network, specifically Graph Attention Network (GAT) (Veličković et al., 2018) which is a commonly-used graph neural network architecture and has shown state-of-the-art performance on various tasks. The GAT computes latent representations of each node in the graph, by attending over its neighbors following a self-attention strategy. To stabilise the learning process of self-attention, the graph attention layer uses multi-head attention as in the Transformer architecture (Vaswani et al., 2017). Namely, the operations of this layer are independently replicated h times (each with different parameters), and outputs are feature-wise concatenated. The inputs to the GAT are, a feature matrix H and an adjacency matrix A. We obtain the adjacency matrix based on the graph construction explained in Section 3.3. We obtain the feature matrix H from the output of the backbone language model as described in Section 3.2. Thus, each node in the graph (token in a document) contains a corresponding embedding. We get an enhanced output representation, $H' = GAT(H, A)$.

We use two variants of our model LAGER$_{\text{nearest}}$ and LAGER$_{\text{angles}}$ based on the two heuristics of graph construction described in Section 3.3.

- **k-nearest neighbors in space:** For this, we use a single GAT ($M = 1$) whose adjacency matrix is based on the k-nearest neighbors in space heuristic.

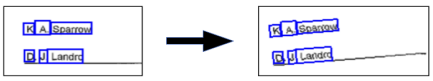
- **k-nearest neighbors at multiple angles:** Based on this heuristic, we construct multiple graphs to gather the spatial information around the token. That is, we construct M graphs that evenly distribute in the space where $M = \lfloor 360°/\theta \rfloor$. For example, if $\theta = 60°$, then we construct 6 graphs for $0°, 60°, 120°, 180°, 240°, 300°$. For each of these M graphs, we use a specific GAT (each with their respective parameters) and then take an average of all the GAT outputs. Specifically,

$$H'_1 = GAT_1(H, A_1) ... H'_M = GAT_M(H, A_M)$$

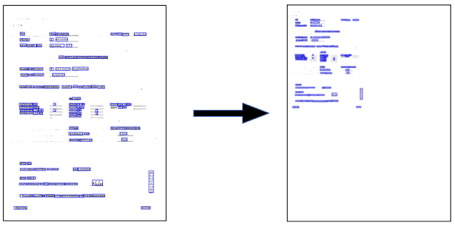
$$H' = \frac{1}{M} \sum H'_i$$

  where A_i is the adjacency matrix constructed with $\frac{i-1}{M} \cdot \lfloor \frac{360°}{\theta} \rfloor$.

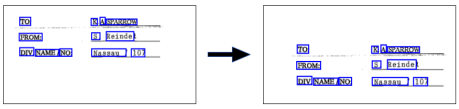
Once we have the embeddings H' from the GAT, it undergoes a linear affine transformation which is represented by the classifier layer in Figure 1. Following this, the model predicts the $\{I, O, B, E, S\}$



(a) Rotation with $\delta = 8°$



(b) Scaling by a factor of 4 ($s_w = 2, s_h = 2$)



(c) Shifting with $a = 10$

Figure 3: Representative examples of the image manipulations

tags for each token in the document and uses sequence labeling to detect each type of entity for the corresponding dataset.

3.5. Image manipulations

In real-world scenarios, document images are often not in ideal, regular conditions and can have some alterations such as shifting, rotation or scaling. As illustrated in Figure 3, we perform three types of manipulations to the document images:

- **Shifting:** In every document, for each bounding box $B = (x_0, y_0, x_1, y_1)$, we translate the coordinates of the four corners with a translation vector (a, a). Thus, the modified bounding box now is $B' = (x_0 + a, y_0 + a, x_1 + a, y_1 + a)$.

- **Rotation:** We rotate each document by a small angle δ about the bottom left corner of the document. Thus, for each bounding box $B = (x_0, y_0, x_1, y_1)$, we rotate the bounding box around (x_0, y_1), i.e. the bottom left corner of the bounding box is the center of rotation. Thus, for every (x, y) corner of a bounding box, we have the resulting rotated point (x', y'), where

$$x' = (x - x_0) \cdot \cos(\delta) - (y - y_1) \cdot \sin(\delta) + x_0$$
$$y' = (x - x_0) \cdot \sin(\delta) + (y - y_1) \cdot \cos(\delta) + y_1$$

- **Scaling:** We scale down, i.e. reduce the size of the entire document by a factor. If w, h denote the width and height of the document and s_w, s_h denote the factor of scaling for the width and height. In every document, for each bounding box $B = (x_0, y_0, x_1, y_1)$, the scaled down coordinates would now be $B = (x_0/s_w, y_0/s_h, x_1/s_w, y_1/s_h)$.

Dataset	# Train Pages	# Test Pages	# Entities / Page
FUNSD	149	50	42.86
CORD	800	100	13.82

Table 1: Dataset Statistics.

4. Experiments

We conduct extensive experiments on the FUNSD (Guillaume Jaume, 2019) and the CORD (Park et al., 2019) datasets under few-shot settings. We also look at how the vanilla baseline models and our proposed models fare under environments where the document images have been manipulated. We also look at some example case studies from both the datasets.

All the experiments under the few-shot settings using few-shot sizes ranging from 1 to 10. We use 6 different random seeds to select the few-shot samples from our training set. We train the different models for a particular few-shot size using the same data and compute the average performance and the standard deviation across the 6 seeds. We report only the result of 2, 3, 4, 5 and 6 shots due to space limitation. We report the results for all 10 few-shot sizes in Table 7 and 8 in the Appendix. For model evaluation, the results are first converted into `IOBES` tagging style and we then compute the word-level precision, recall and F-1 score using the seqeval APIs (Nakayama, 2018). All implementation details including hyperparameters used for all the experiments is in Section 4.3.

4.1. Datasets

Our experiments are conducted on two real-world data collections: FUNSD and CORD. Both datasets provide rich annotations for the document images and include the words and the word-level bounding boxes. The details and statistics (Table 1) of these two datasets are as follows.

- **FUNSD:** FUNSD consists of 199 fully-annotated, noisy-scanned forms with various appearance and format which makes the form understanding task more challenging. The word spans in this datasets are annotated with three different labels: `header`, `question` and `answer`, and the rest words are annotated as `other`. We use the original label names.
- **CORD:** CORD consists of about 1000 receipts with annotations of bounding boxes and textual contents. The word spans in this datasets are annotated with 30 different labels. The broad categories include `menu`, `sub-total` and `total`. We use the original label names.

	$\|\mathcal{P}\|$	Model	FUNSD			CORD		
			Precision	Recall	F-1	Precision	Recall	F-1
LayoutLMv2	2	Vanilla	38.61±4.04	53.8±8.35	44.82±5.3	43.34±2.07	55.85±2.43	48.66±2.13
		+ LAGER$_{nearest}$	41.4±3.26	52.08±9.89	45.74±5.32	43.42±2.45	55.89±3.15	48.87±2.73
		+ LAGER$_{angles}$	**41.45±2.22**	**54.21±2.99**	**46.9±1.66**	**43.77±2.77**	**56.22±2.94**	**49.21±2.8**
	3	Vanilla	47.73±2.91	61.16±7.02	53.55±4.36	50.52±2.87	61.03±3.8	54.69±3.19
		+ LAGER$_{nearest}$	**48.72±3.33**	**63.12±8.69**	**54.9±5.42**	**50.81±3.03**	**61.49±3.48**	**55.02±3.09**
		+ LAGER$_{angles}$	48.48±3.46	60.68±9.14	53.79±5.73	50.15±2.79	61.28±2.98	54.53±2.7
	4	Vanilla	51.13±2.16	64.57±6.8	56.95±3.7	55.18±2.25	65.96±3.35	59.88±2.66
		+ LAGER$_{nearest}$	**52.59±3.71**	**67.23±4.75**	**58.93±3.55**	54.86±2.57	65.87±3.88	59.85±3.05
		+ LAGER$_{angles}$	50.86±3.43	66.47±4.56	57.59±3.63	**55.52±2.56**	**66.57±3.45**	**60.54±2.89**
	5	Vanilla	53.46±2.43	64.57±6.71	58.36±3.63	57.59±3.11	68.11±3.24	62.27±3.06
		+ LAGER$_{nearest}$	**54.0±2.54**	**67.77±4.98**	**60.02±2.83**	57.49±3.1	68.27±2.55	62.41±2.82
		+ LAGER$_{angles}$	52.98±3.25	67.23±5.08	59.19±3.53	**57.88±2.73**	**68.27±2.48**	**62.63±2.52**
	6	Vanilla	56.92±1.59	67.59±2.9	61.77±1.77	60.28±2.47	70.01±2.48	64.41±2.43
		+ LAGER$_{nearest}$	57.45±2.57	71.14±2.83	63.53±2.25	**60.63±2.88**	**70.49±2.69**	**65.19±2.78**
		+ LAGER$_{angles}$	**58.14±3.79**	**71.34±3.02**	**63.95±2.21**	59.97±2.37	70.04±2.28	64.6±2.26

Table 2: Evaluation results with LayoutLMv2 as the backbone model on different few-shot sizes. **Bold** denotes the best model

	$\|\mathcal{P}\|$	Model	FUNSD			CORD		
			Precision	Recall	F-1	Precision	Recall	F-1
LayoutLMv3	2	Vanilla	44.29±6.14	58.96±7.2	50.43±6.03	47.21±6.25	58.99±4.94	52.41±5.85
		+ LAGER$_{nearest}$	**49.82±6.06**	**59.55±8.91**	**54.09±6.54**	**48.68±5.72**	60.19±4.23	**53.79±5.24**
		+ LAGER$_{angles}$	46.8±6.46	58.15±8.92	51.61±6.42	48.15±5.07	**60.3±3.57**	53.51±4.58
	3	Vanilla	59.66±4.92	72.2±7.65	65.29±5.92	51.34±6.55	62.49±5.47	56.34±6.2
		+ LAGER$_{nearest}$	62.18±5.13	**73.12±7.3**	**67.12±5.56**	**53.08±7.32**	**64.3±5.55**	**58.1±6.73**
		+ LAGER$_{angles}$	**60.73±5.09**	72.41±7.64	65.97±5.71	52.77±7.17	63.72±5.55	57.68±6.63
	4	Vanilla	65.32±3.89	77.97±2.26	71.06±3.04	54.18±5.01	64.92±3.76	59.04±4.53
		+ LAGER$_{nearest}$	**67.86±3.3**	**78.73±2.57**	**72.86±2.69**	**56.28±4.24**	**66.47±3.29**	**60.94±3.86**
		+ LAGER$_{angles}$	65.93±3.28	77.22±3.45	71.08±2.81	55.38±4.63	65.99±3.79	60.21±4.3
	5	Vanilla	67.14±5.17	77.88±2.62	72.07±4.01	58.55±2.82	67.03±2.46	62.49±2.57
		+ LAGER$_{nearest}$	69.6±2.57	79.72±1.66	74.3±1.94	**59.84±3.27**	68.36±2.34	**63.8±2.76**
		+ LAGER$_{angles}$	**70.32±1.41**	**80.86±1.23**	**75.22±1.1**	59.37±4.09	**68.48±3.08**	63.58±3.63
	6	Vanilla	71.19±3.75	80.83±1.09	75.68±2.58	60.91±3.51	69.16±2.76	64.76±3.16
		+ LAGER$_{nearest}$	72.71±3.42	81.53±1.98	**76.84±2.58**	**61.8±5.14**	70.0±3.75	65.63±4.53
		+ LAGER$_{angles}$	**72.31±3.7**	**81.65±1.81**	76.67±2.7	61.56±4.49	**70.3±3.33**	**65.63±3.98**

Table 3: Evaluation results with LayoutLMv3 as the backbone model on different few-shot sizes. **Bold** denotes the best model

4.2. Baselines

Based on Table 6 in the Appendix, we select the two strongest baselines that are representative for our task, i.e. LayoutLMv2 and LayoutLMv3. In our model LAGER, we use LayoutLMv2 (Xu et al., 2021) and LayoutLMv3 (Huang et al., 2022) as backbones . We evaluate LAGER against vanilla LayoutLMv2 and LayoutLMv3 in few-shot setting.

- **LayoutLMv2:** is a multi-modal language model which is an improved version of LayoutLM (Xu et al., 2020). It integrates the visual information in the pre-training stage to learn the cross-modality interaction between visual and textual information.
- **LayoutLMv3:** is another large multi-modal language model which aims to mitigate the discrepancy between text and image modalities in other models such as LayoutLM and LayoutLMv2. It facilitates multimodal representation learning by unifying the text and image masking.

For all our experiments we use the base version of the models and follow the `IOBES` tagging scheme.

4.3. Implementation Details

We build our model on top of LayoutLMv2/LayoutLMv3 as our backbone language model. We use the Transformers (Wolf et al., 2019) and also utilize the repository of Dong et al. (2019) to build our model. We use one NVIDIA A6000 to finetune with batch size of 8. We optimize the model with AdamW optimizer and the learning rate is 5×10^{-5}.

We ran extensive experiments for various intuitive choices of hyperparameters. For the value of k during graph construction, we try different values like 1,2,4, and 8. All results reported for both heuristics use $k = 4$. For the k-nearest neighbors at multiple angles, the idea is to capture the topological relationship of a token. Thus, it's quite natural to divide the 2D plane into 2,4,6 or 8 halves, i.e. angles such as 90, 60, 45, 30, 15, etc. were tried. Though most of these choices work great, the one in the experiments reported use $\theta = 60°$. Thus,

	$\|\mathcal{P}\|$	Model	Shift ($a = 20$)				Scale ($s_w = 2$, $s_h = 2$)				Rotation ($\delta = 8°$)			
			FUNSD		CORD		FUNSD		CORD		FUNSD		CORD	
			F-1	**Diff.**	**F-1**	**Diff.**	**F-1**	**Diff.**	**F-1**	**Diff.**	**F-1**	**Diff.**	**F-1**	**Diff.**
LayoutLMv3	2	Vanilla	49.28±5.69	1.15	50.53±5.58	1.88	32.66±15.64	17.77	38.77±6.62	13.64	48.11±5.77	2.33	48.39±5.31	4.02
		+ LAGER$_{\text{nearest}}$	**53.31±5.03**	0.78	51.97±5.24	1.82	**38.07±16.16**	16.02	**40.66±7.63**	**13.13**	**52.56±6.22**	**1.53**	50.4±5.48	3.39
		+ LAGER$_{\text{angles}}$	51.32±5.83	**0.29**	**51.98±4.45**	**1.53**	36.14±15.32	**15.47**	39.96±7.98	13.55	49.58±6.64	2.03	**50.3±4.97**	**3.21**
	3	Vanilla	63.22±5.5	2.07	54.15±5.76	2.19	46.44±15.39	18.85	43.04±7.23	13.3	63.24±5.69	2.05	51.2±5.72	5.14
		+ LAGER$_{\text{nearest}}$	**65.34±4.58**	**1.78**	**55.94±6.22**	2.16	48.78±12.16	18.34	44.6±8.52	13.5	**65.36±5.46**	**1.76**	**52.84±6.71**	5.25
		+ LAGER$_{\text{angles}}$	63.98±4.37	1.99	55.72±5.96	**1.96**	**49.61±11.33**	**16.36**	**44.63±7.46**	**13.05**	64.04±5.65	1.92	52.72±6.56	**4.96**
	4	Vanilla	68.66±3.13	2.4	56.8±3.84	2.24	52.7±9.16	18.36	44.17±5.89	14.47	68.84±3.34	**2.22**	54.57±4.24	4.47
		+ LAGER$_{\text{nearest}}$	**70.62±2.81**	2.24	**58.97±3.35**	**1.97**	**54.86±9.4**	**18**	**46.64±6.54**	**14.3**	**70.4±3.06**	2.46	**56.9±3.17**	**4.04**
		+ LAGER$_{\text{angles}}$	68.92±2.66	**2.16**	58.06±3.87	2.15	52.99±11.59	18.09	45.86±6.12	14.35	68.71±3.0	2.36	55.76±3.63	4.45
	5	Vanilla	70.01±4.08	2.06	59.47±3.23	3.02	47.18±15.26	24.89	45.02±4.41	17.47	69.62±4.13	2.45	57.31±2.85	5.18
		+ LAGER$_{\text{nearest}}$	72.43±1.66	**1.87**	**61.73±2.52**	**2.07**	**57.18±8.11**	**17.12**	**49.83±3.18**	**13.97**	72.36±2.66	**1.93**	**58.88±2.33**	**4.91**
		+ LAGER$_{\text{angles}}$	**73.28±1.15**	1.94	61.01±3.15	2.57	56.73±5.68	18.49	48.55±3.08	15.03	**72.86±2.1**	2.35	58.33±2.54	5.25
	6	Vanilla	73.58±1.43	2.1	61.91±3.18	2.85	53.07±13.76	22.61	47.18±1.7	17.58	73.54±3.15	2.14	60.19±2.04	4.57
		+ LAGER$_{\text{nearest}}$	74.59±2.79	2.25	**63.35±4.41**	**2.28**	**57.0±11.24**	**19.84**	**52.34±3.02**	**13.29**	75.09±2.75	**1.75**	**61.26±3.25**	**4.37**
		+ LAGER$_{\text{angles}}$	**75.07±2.52**	**1.6**	63.24±3.64	2.39	56.53±12.75	20.14	51.68±4.05	13.95	**74.72±2.94**	1.95	60.93±2.67	4.69

Table 4: Evaluation results on image manipulation with shifting ($a = 20$), scaling by a factor of 4 ($s_w = 2$,$s_h = 2$) and rotation with $\delta = 8°$ with LayoutLMv3 as backbone. The column Diff. refers to the difference in F1-scores between results in setting without manipulation (Table 3) and with manipulation.
Bold denotes the best model.

we use $M = 360°/60° = 6$ GATs and average the outputs of these different GATs when we run the experiments. For all our experiments, we set the number of heads in the GAT, to $h = 4$.

4.4. Few-shot Experimental results

We report our results using the two baseline models described in Section 4.2 in Tables 2 and 3. For the baseline and as the backbone language model in LAGER, Table 2 and 3 use LayoutLMv2 and LayoutLMv3 respectively. The results are reported on two versions of our model, LAGER$_{\text{nearest}}$ and LAGER$_{\text{angles}}$ for the two heuristics of graph construction described in Section 3.3 and 3.4. We observe that our model achieves significant performance improvements compared with the baselines for both FUNSD and CORD datasets. We see in Table 2, there is on average relative improvements of 4% and 1.5% in terms of F-1 score for FUNSD and CORD respectively over the vanilla LayoutLMv2 baseline. For Table 3, we see an average relative improvement in terms of F-1 score by 4% and 3% for FUNSD and CORD respectively over the LayoutLMv3 baseline. We see similar gains in performance for precision and recall in both the tables.

We also analyze the filewise results of the test set instances for both FUNSD and CORD. That is, for each individual test set instance, we compare the filewise F-1 scores of our models with the baseline. We observe that when using LayoutLMv2 as the backbone, our models on average improve over the baseline for 58% and 62% of our test set instances for FUNSD and CORD respectively. Similarly, when using LayoutLMv3 as the backbone, our models on average improve over the baseline for 65% and 67% of our test set instances for FUNSD and CORD respectively. This shows that LAGER$_{\text{nearest}}$ and LAGER$_{\text{angles}}$ provide more confident predictions leading to the overall performance improvement for the entity recognition task.

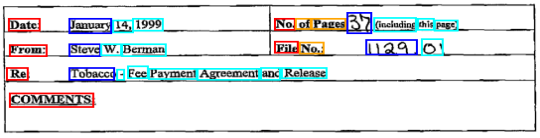
Based on these comparisons, we conclude that our proposed framework is superior to the traditional vanilla language model baselines in few-shot settings.

4.5. Experiments with image manipulation

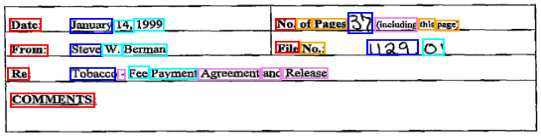
In these experiments, for the models in Table 3 we manipulate the test-set images during inference as described in Section 3.5. We perform experiments with various factors of shifting, scaling and rotation and observe similar evaluation results. We show an instance of each here due to space constraints. We use shifting with a factor of $a = 20$, scaling with a factor of 4, i.e $s_h = 2, s_w = 2$ and rotation with $\delta = 8°$. The results are reported in Table 4. We see that for both the scaling and shifting, both LAGER$_{\text{nearest}}$ and LAGER$_{\text{angles}}$ approaches perform better than the vanilla LayoutLMv3 baseline for all cases. As expected, we see a drop in performance in all the models. We measure the difference in F-1 scores between results without manipulation (Table 3) and with manipulation. From these numbers, we see that for all models, the difference for each few-shot size is lower for both our approaches than the baselines for both FUNSD and CORD. This shows that our method is more robust compared to the baseline to these manipulations.

4.6. Case studies

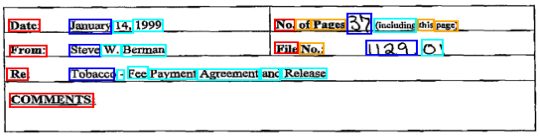
We visualize several cases from the 4-shot setting. In Figure 4, the models use LayoutLMv2 as backbone and we show an example from the FUNSD test set. We observe that compared to the



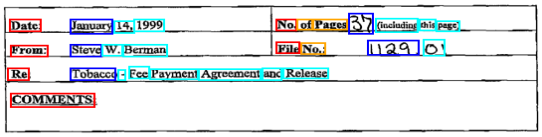
(a) Ground truth



(b) LayoutLMv2

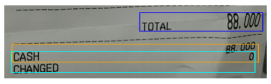


(c) LayoutLMv2 + LAGER$_{\text{nearest}}$

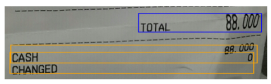


(d) LayoutLMv2 + LAGER$_{\text{angles}}$

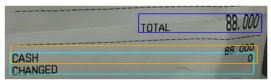
Figure 4: Case studies from FUNSD. , , , , denote `B-ANSWER`, `I-ANSWER`, `B-QUESTION`, `I-QUESTION` and `OTHER` respectively.



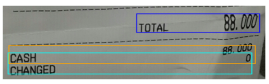
(a) Ground truth



(b) LayoutLMv3



(c) LayoutLMv3 + LAGER$_{\text{nearest}}$



(d) LayoutLMv3 + LAGER$_{\text{angles}}$

Figure 5: Case studies from CORD. , , , denote the tags for `TOTAL.TOTAL_PRICE` `TOTAL.CASHPRICE` and `TOTAL.CHANGEPRICE` respectively.

ground-truth, the vanilla LayoutLMv2 makes errors specifically when there are a sequence of tokens next to each other all with the `I-ANSWER` tag. We see that both the approaches LAGER$_{\text{nearest}}$ and LAGER$_{\text{angles}}$ are able to capture the continuous set of words in the form correctly. We believe that our graph based approach is able to capture the spatial relationship of these words and is thus able to get better predictions. Further, the LAGER$_{\text{angles}}$ approach also captures the mislabeled `I-QUESTION` tags by LAGER$_{\text{nearest}}$. We show another example in Figure 5 in which we use LayoutLMv3 as the backbone and show an example from the CORD test set. We see that compared to the ground truth, the vanilla LayoutLMv3 model misclassifies the `TOTAL-CHANGEPRICE` tag. We see that both the approaches LAGER$_{\text{nearest}}$ and LAGER$_{\text{angles}}$ are able to classify that correctly.

5. Conclusion and Future Work

We present LAGER, a layout-aware graph based entity recognition framework for few-shot entity recognition in document images. Existing methods use the coordinates of the token bounding boxes to encode layout information and they are sensitive to manipulations in the images such as shifting, rotation or scaling especially in low data resource settings. Our approach makes use of the topological relationship between the tokens in the documents by using a graph-based approach and it is more robust to these manipulations. We construct graphs based on heuristics relating to the k-nearest neighbors of these tokens in space and at a certain angle. We extend layout-aware pre-trained language models with a graph attention network with the graphs we construct and the output hidden states froms the backbone language model. Extensive experiments in few-shot settings on FUNSD and CORD datasets illustrate the performance gains using our approach. Further, we show experiments with image manipulations where our approach is robust to these alterations in the image. In the future, we plan to apply the model on other backbones and incorporate other features such as the semantic relationship between the tokens in addition to the topological relationship when constructing the graph.

Limitations

The density of the graph constructed in terms of edge connectivity is dependent on the layout of the tokens present in the document. This leads to certain types of documents or even certain documents within a dataset to have a very dense graph whilst other documents can have sparse graphs. This could be a factor that affects the output representations from the GAT and the performance of the model.

Ethics Statement

Our work focuses on few-shot entity recognition in document images. Both the datasets that we use are public and builds upon language models that are open-source. We also plan to release our code publicly. Thus, we do not anticipate any ethical concerns.

6. Bibliographical References

Alfred V. Aho and Jeffrey D. Ullman. 1972. *The Theory of Parsing, Translation and Compiling*, volume 1. Prentice-Hall, Englewood Cliffs, NJ.

American Psychological Association. 1983. *Publications Manual*. American Psychological Association, Washington, DC.

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *Journal of Machine Learning Research*, 6:1817–1853.

Galen Andrew and Jianfeng Gao. 2007. Scalable training of L_1-regularized log-linear models. In *Proceedings of the 24th International Conference on Machine Learning*, pages 33–40.

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R. Manmatha. 2021. Docformer: End-to-end transformer for document understanding.

BSI. 1973a. *Natural Fibre Twines*, 3rd edition. British Standards Institution, London. BS 2570.

BSI. 1973b. Natural fibre twines. BS 2570, British Standards Institution, London. 3rd. edn.

A. Castor and L. E. Pollux. 1992. The use of user modelling to guide inference and learning. *Applied Intelligence*, 2(1):37–53.

Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. 1981. Alternation. *Journal of the Association for Computing Machinery*, 28(1):114–133.

J.L. Chercheur. 1994. *Case-Based Reasoning*, 2nd edition. Morgan Kaufman Publishers, San Mateo, CA.

N. Chomsky. 1973. Conditions on transformations. In *A festschrift for Morris Halle*, New York. Holt, Rinehart & Winston.

James W. Cooley and John W. Tukey. 1965. An algorithm for the machine calculation of complex Fourier series. *Mathematics of Computation*, 19(90):297–301.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Li Dong, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon. 2019. Unified language model pre-training for natural language understanding and generation. *arXiv preprint arXiv:1905.03197*.

Umberto Eco. 1990. *The Limits of Interpretation*. Indian University Press.

Jean-Philippe Thiran Guillaume Jaume, Hazim Kemal Ekenel. 2019. Funsd: A dataset for form understanding in noisy scanned documents. In *Accepted to ICDAR-OST*.

Dan Gusfield. 1997. *Algorithms on Strings, Trees and Sequences*. Cambridge University Press, Cambridge, UK.

Paul Gerhard Hoel. 1971a. *Elementary Statistics*, 3rd edition. Wiley series in probability and mathematical statistics. Wiley, New York, Chichester. ISBN 0 471 40300.

Paul Gerhard Hoel. 1971b. *Elementary Statistics*, 3rd edition, Wiley series in probability and mathematical statistics, pages 19–33. Wiley, New York, Chichester. ISBN 0 471 40300.

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 *Proceedings of the 30th ACM International Conference on Multimedia*.

Otto Jespersen. 1922. *Language: Its Nature, Development, and Origin*. Allen and Unwin.

Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. 2022. Formnet: Structural encoding beyond sequential modeling in form document information extraction. In *Annual Meeting of the Association for Computational Linguistics*.

Chen-Yu Lee, Chun-Liang Li, Hao Zhang, Timothy Dozat, Vincent Perot, Guolong Su, Xiang Zhang, Kihyuk Sohn, Nikolai Glushnev, Renshen Wang, et al. 2023. Formnetv2: Multimodal graph contrastive learning for form document information extraction. *arXiv preprint arXiv:2305.02549*.

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

Colin Lockard, Prashant Shiralkar, and Xin Dong. 2019. Openceres: When open information extraction meets the semi-structured web. In *North American Chapter of the Association for Computational Linguistics*.

Colin Lockard, Prashant Shiralkar, Xin Luna Dong, and Hannaneh Hajishirzi. 2020. ZeroShotCeres: Zero-shot relation extraction from semi-structured webpages. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8105–8117, Online. Association for Computational Linguistics.

Jaya Krishna Mandivarapu, Eric bunch, and Glenn fung. 2021. Domain agnostic few-shot learning for document intelligence.

Lluis Marquez, Pere Comas, Jesús Giménez, and Neus Catala. 2005. Semantic role labeling as sequential tagging. In *Proceedings of the Ninth Conference on Computational Natural Language Learning (CoNLL-2005)*, pages 193–196.

Hiroki Nakayama. 2018. seqeval: A python framework for sequence labeling evaluation. Software available from https://github.com/chakki-works/seqeval.

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.

Mohammad Sadegh Rasooli and Joel R. Tetreault. 2015. Yara parser: A fast and accurate dependency parser. *Computing Research Repository*, arXiv:1503.06733. Version 2.

Lev Ratinov and Dan Roth. 2009. Design challenges and misconceptions in named entity recognition. In *Proceedings of the Thirteenth Conference on Computational Natural Language Learning (CoNLL-2009)*, pages 147–155.

Charles Joseph Singer, E. J. Holmyard, and A. R. Hall, editors. 1954–58. *A history of technology*. Oxford University Press, London. 5 vol.

Jannik Strötgen and Michael Gertz. 2012. Temporal tagging on different domains: Challenges, strategies, and gold standards. In *Proceedings of the Eight International Conference on Language Resources and Evaluation (LREC'12)*, pages 3746–3753, Istanbul, Turkey. European Language Resource Association (ELRA).

S. Superman, B. Batman, C. Catwoman, and S. Spiderman. 2000. *Superheroes experiences with books*, 20th edition. The Phantom Editors Associates, Gotham City.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. 2018. Graph Attention Networks. *International Conference on Learning Representations*.

Zilong Wang, Jiuxiang Gu, Chris Tensmeyer, Nikolaos Barmpalios, Ani Nenkova, Tong Sun, Jingbo Shang, and Vlad I. Morariu. 2022. Mgdoc: Pretraining with multi-granular hierarchy for document image understanding.

Zilong Wang and Jingbo Shang. 2022. Towards few-shot entity recognition in document images: A label-aware sequence-to-sequence framework. In *Findings of the Association for Computational Linguistics: ACL 2022*, Dublin, Ireland. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, et al. 2019. Huggingface's transformers: State-of-the-art natural language processing. *arXiv preprint arXiv:1910.03771*.

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. 2021. Layoutlmv2: Multi-modal pretraining for visually-rich document understanding. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL) 2021*.

Yiheng Xu, Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, and Ming Zhou. 2020. Layoutlm: Pretraining of text and layout for document image understanding. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1192–1200.

Shi-Xue Zhang, Xiaobin Zhu, Jie-Bo Hou, Chang Liu, Chun Yang, Hongfa Wang, and Xu-Cheng Yin. 2020. Deep relational reasoning graph network for arbitrary shape text detection. In *IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*.

7. Appendix

7.1. Baseline models

There are several popular open source models in this domain: re BERT (Devlin et al., 2018), RoBERTa (Liu et al., 2019), LayoutLM(Xu et al., 2020), LayoutLMv2(Xu et al., 2021) and LayoutLMv3(Huang et al., 2022).

- **BERT**(Devlin et al., 2018) is a text-only auto-encoding pre-trianed language model that uses masked language modeling and next sentence prediction as its pre-training tasks. For this task, we fine-tune the pre-trained BERT base model with the few-shot training samples for both the datasets.

- **RoBERTa**(Liu et al., 2019) is an extension of BERT that is trained on more data and also makes modifications to its pre-training tasks thereby achieving better performance in numerous natural language understanding tasks. Similar to BERT, we fine-tune the base model with the few-shot training samples for both the datasets.

- **LayoutLM**(Xu et al., 2020) is a multimodal language model that includes layout and text information. LayoutLM is built upon BERT and adds extra spatial embeddings into the BERT embedding layer.

- **LayoutLMv2**(Xu et al., 2021) is a multi-modal language model which is an improved version of LayoutLM (Xu et al., 2020). It integrates the visual information in the pre-training stage to learn the cross-modality interaction between visual and textual information.

- **LayoutLMv3**(Huang et al., 2022) is another large multi-modal language model which aims to mitigate the discrepancy between text and image modalities in other models such as LayoutLM and LayoutLMv2. It facilitates multi-modal representation learning by unifying the text and image masking.

From Table 6 we can see that LayoutLMv2 and LayoutLMv3 are the two strongest models and significantly outperform BERT, RoBERTa and LayoutLM. Thus, in our LAGER framework, we perform our experiments by picking LayoutLmv2 or LayoutLMv3 as our pre-trained layout aware backbone language model.

Table 7 and 8 comprises of few-shot experimental results using LayoutLMv2 and LayoutLMv3 as the backbone models respectively for few-shot sizes from 1 to 10.

Model	FUNSD-F1	CORD-F1
LayoutLMv3	89.92	81.95
LayoutLMv3 + $\text{LAGER}_{\text{nearest}}$	90.23	**82.14**
LayoutLMv3 + $\text{LAGER}_{\text{angles}}$	**90.34**	82.09

Table 5: Evaluation results with LayoutLMv3 as the backbone model using the entire dataset. **Bold** denotes the best model

7.2. Experiments with entire datasets

Though the model that we construct isn't tailored for a few-shot setting, we believe our approach is particularly useful in a few-shot setting when there is limited data availability for entity recognition, as the graph-based method is efficient to train and easier to generalize. We perform additional experiments using the entire dataset to validate if the graph based approach is effective even when trained with the entire dataset and not in a few-shot setting. We run experiments using LayoutLMv3 as the backbone as shown in Table 5 and observe improvements in both of our approaches, $\text{LAGER}_{\text{nearest}}$ and $\text{LAGER}_{\text{angles}}$. However, our main contribution is for the few-shot setting. Our approach focuses on few-shot settings and limited data availability scenarios where we have potentially a larger number of documents for testing.

$\mathcal{P}$	Model	FUNSD			CORD		
		Precision	Recall	F-1	Precision	Recall	F-1
1	BERT	10.93±3.63	22.39±6.68	13.97±2.84	22.4±4.91	31.24±6.4	26.08±5.55
	RoBERTa	12.98±4.34	21.93±11.09	17.13±6.07	18.55±6.69	25.77±9.24	21.56±7.73
	LayoutLM	15.77±5.35	23.03±7.52	22.03±5.15	26.5±9.06	35.58±11.17	30.37±10.02
	LayoutLMv2	**23.4±10.05**	**29.22±14.6**	**25.51±11.72**	**32.5±4.15**	**42.51±3.82**	**36.33±4.03**
	LayoutLMv3	**22.93±6.21**	**39.7±5.94**	**28.85±6.35**	**36.56±5.46**	**47.58±6.44**	**41.33±5.88**
2	BERT	15.51±2.29	28.14±4.02	19.74±2.05	30.05±5.93	41.63±6.02	34.87±6.12
	RoBERTa	21.64±1.64	33.43±4.24	26.68±1.76	34.96±6.73	45.7±7.17	39.59±7.03
	LayoutLM	33.05±4.85	35.52±8.81	28.02±6.07	38.51±7.88	50.52±6.81	43.63±7.66
	LayoutLMv2	**38.61±4.04**	**53.8±8.35**	**44.82±5.3**	**43.34±2.07**	**55.85±2.43**	**48.66±2.13**
	LayoutLMv3	**44.29±6.14**	**58.96±7.2**	**50.43±6.03**	**47.21±6.25**	**58.99±4.94**	**52.41±5.85**
3	BERT	19.42±3.75	32.63±5.62	24.3±4.44	32.57±8.07	44.9±8.73	37.72±8.53
	RoBERTa	25.22±3.22	39.0±5.37	30.57±3.76	41.0±8.37	51.07±8.35	45.46±8.48
	LayoutLM	28.69±3.86	46.07±8.95	35.13±7.29	43.35±6.77	56.15±4.62	48.84±6.11
	LayoutLMv2	**47.73±2.91**	**61.16±7.02**	**53.55±4.36**	**50.52±2.87**	**61.03±3.8**	**54.69±3.19**
	LayoutLMv3	**59.66±4.92**	**72.2±7.65**	**65.29±5.92**	**51.34±6.55**	**62.49±5.47**	**56.34±6.2**
4	BERT	21.2±3.54	37.04±3.13	26.9±3.59	36.48±8.43	48.17±8.47	41.47±8.64
	RoBERTa	27.53±2.92	42.83±2.68	33.48±2.83	45.89±7.84	55.04±8.69	50.05±8.25
	LayoutLM	34.31±2.56	52.23±5.45	41.29±2.68	48.41±6.28	60.5±4.25	53.7±5.58
	LayoutLMv2	**51.13±2.16**	**64.57±6.8**	**56.95±3.7**	**55.18±2.25**	**65.96±3.35**	**59.88±2.66**
	LayoutLMv3	**65.32±3.89**	**77.97±2.26**	**71.06±3.04**	**54.18±5.01**	**64.92±3.76**	**59.04±4.53**
5	BERT	24.2±3.24	39.59±2.55	29.97±3.0	37.75±8.26	49.53±8.19	42.81±8.37
	RoBERTa	31.57±2.56	46.77±2.14	37.65±2.2	48.51±8.28	57.32±10.11	52.54±9.07
	LayoutLM	38.60±5.12	54.07±5.49	44.87±4.61	52.05±5.68	63.7±3.79	57.23±4.95
	LayoutLMv2	**53.46±2.43**	**64.57±6.71**	**58.36±3.63**	**57.59±3.11**	**68.11±3.24**	**62.27±3.06**
	LayoutLMv3	**67.14±5.17**	**77.88±2.62**	**72.07±4.01**	**58.55±2.82**	**67.03±2.46**	**62.49±2.57**
6	BERT	26.54±1.99	41.47±3.69	32.27±1.99	42.04±5.46	54.18±4.43	47.31±5.13
	RoBERTa	33.75±2.19	47.2±2.54	39.32±2.06	52.88±4.84	61.41±4.86	56.82±4.82
	LayoutLM	42.27±3.85	57.84±4.49	48.79±3.75	52.05±5.68	63.7±3.79	57.23±4.95
	LayoutLMv2	**56.92±1.59**	**67.59±2.9**	**61.77±1.77**	**60.28±2.47**	**70.01±2.48**	**64.41±2.43**
	LayoutLMv3	**71.19±3.75**	**80.83±1.09**	**75.68±2.58**	**60.91±3.51**	**69.16±2.76**	**64.76±3.16**
7	BERT	28.67±1.94	44.04±4.07	34.59±1.64	43.31±5.46	54.8±4.5	48.35±5.15
	RoBERTa	35.27±2.7	49.24±4.49	41.01±2.77	54.95±3.64	62.66±4.0	58.55±3.78
	LayoutLM	45.81±2.59	61.24±4.05	52.29±1.93	57.9±2.22	67.88±1.96	62.48±1.95
	LayoutLMv2	**59.43±3.59**	**68.98±3.73**	**63.71±2.19**	**60.71±2.09**	**69.95±2.16**	**64.86±2.03**
	LayoutLMv3	**72.44±3.56**	**81.56±1.12**	**76.68±1.95**	**62.26±3.78**	**70.3±3.02**	**66.03±3.43**
8	BERT	30.81±2.83	43.72±3.99	36.11±3.12	45.58±5.27	57.34±4.27	50.76±4.94
	RoBERTa	37.3±3.55	49.52±4.89	42.52±3.93	57.38±1.86	65.32±1.54	61.08±1.57
	LayoutLM	48.48±3.09	60.21±4.66	53.68±3.51	57.9±2.22	67.88±1.96	62.48±1.95
	LayoutLMv2	**61.69±2.93**	**69.9±3.3**	**65.51±2.75**	**62.98±0.94**	**72.07±1.24**	**67.18±1.07**
	LayoutLMv3	**74.31±2.19**	**81.75±2.6**	**77.85±2.29**	**64.49±3.24**	**72.21±2.17**	**68.12±2.77**
9	BERT	31.18±2.75	43.67±5.27	36.33±3.51	47.25±3.93	59.11±2.9	52.5±3.54
	RoBERTa	37.3±3.41	49.74±4.26	42.6±3.61	58.77±2.22	66.52±1.09	62.39±1.62
	LayoutLM	51.91±2.37	63.59±4.04	57.14±2.95	57.9±2.22	67.88±1.96	62.48±1.95
	LayoutLMv2	**62.54±2.22**	**71.17±3.65**	**66.55±2.54**	**63.93±0.5**	**72.63±0.63**	**67.96±0.15**
	LayoutLMv3	**75.9±1.53**	**82.52±1.36**	**79.06±1.24**	**65.89±2.8**	**73.44±1.87**	**69.45±2.39**
10	BERT	32.32±3.55	45.16±5.04	37.59±3.84	50.83±3.11	61.91±2.25	55.81±2.76
	RoBERTa	38.65±3.64	51.1±4.77	43.98±3.93	60.22±2.28	67.73±1.59	63.75±1.9
	LayoutLM	52.94±2.51	64.29±3.3	58.05±2.63	64.05±2.76	71.99±1.91	67.78±2.33
	LayoutLMv2	**63.49±2.7**	**72.97±2.21**	**67.89±2.33**	**66.18±0.99**	**73.79±0.8**	**69.92±0.76**
	LayoutLMv3	**75.9±1.53**	**82.52±1.36**	**79.06±1.24**	**65.89±2.8**	**73.44±1.87**	**69.45±2.39**

Table 6: Comparison of various baseline models for all few-shot sizes. **Bold** denotes the best two models

	$\mathcal{P}$	Model	FUNSD Precision	FUNSD Recall	FUNSD F-1	CORD Precision	CORD Recall	CORD F-1
LayoutLMv2	1	Vanilla	23.4±10.05	29.22±14.6	25.51±11.72	32.5±4.15	42.51±3.82	36.33±4.03
		+ LAGER$_{nearest}$	**26.66±4.36**	**40.56±7.7**	**32.08±5.35**	**31.75±6.3**	**43.55±7.56**	**36.36±6.89**
		+ LAGER$_{angles}$	26.58±5.06	36.62±13.16	29.72±8.13	30.56±6.63	41.37±8.34	35.14±7.39
	2	Vanilla	38.61±4.04	53.8±8.35	44.82±5.3	43.34±2.07	55.85±2.43	48.66±2.13
		+ LAGER$_{nearest}$	41.4±3.26	52.08±9.89	45.74±5.32	43.42±2.45	55.89±3.15	48.87±2.73
		+ LAGER$_{angles}$	**41.45±2.22**	**54.21±2.99**	**46.9±1.66**	**43.77±2.77**	**56.22±2.94**	**49.21±2.8**
	3	Vanilla	47.73±2.91	61.16±7.02	53.55±4.36	50.52±2.87	61.03±3.8	54.69±3.19
		+ LAGER$_{nearest}$	**48.72±3.33**	**63.12±8.69**	**54.9±5.42**	**50.81±3.03**	**61.49±3.48**	**55.02±3.09**
		+ LAGER$_{angles}$	48.48±3.46	60.68±9.14	53.79±5.73	50.15±2.79	61.28±2.98	54.53±2.7
	4	Vanilla	51.13±2.16	64.57±6.8	56.95±3.7	55.18±2.25	65.96±3.35	59.88±2.66
		+ LAGER$_{nearest}$	**52.59±3.71**	**67.23±4.75**	**58.93±3.55**	54.86±2.57	65.87±3.88	59.85±3.05
		+ LAGER$_{angles}$	50.86±3.43	66.47±4.56	57.59±3.63	**55.52±2.56**	**66.57±3.45**	**60.54±2.89**
	5	Vanilla	53.46±2.43	64.57±6.71	58.36±3.63	57.59±3.11	68.11±3.24	62.27±3.06
		+ LAGER$_{nearest}$	**54.0±2.54**	**67.77±4.98**	**60.02±2.83**	57.49±3.1	68.27±2.55	62.41±2.82
		+ LAGER$_{angles}$	52.98±3.25	67.23±5.08	59.19±3.53	**57.88±2.73**	**68.27±2.48**	**62.63±2.52**
	6	Vanilla	56.92±1.59	67.59±2.9	61.77±1.77	60.28±2.47	70.01±2.48	64.41±2.43
		+ LAGER$_{nearest}$	57.45±2.57	71.14±2.83	63.53±2.25	**60.63±2.88**	**70.49±2.69**	**65.19±2.78**
		+ LAGER$_{angles}$	**58.14±3.79**	**71.34±3.02**	**63.95±2.21**	59.97±2.37	70.04±2.28	64.6±2.26
	7	Vanilla	59.43±3.59	68.98±3.73	63.71±2.19	60.71±2.09	69.95±2.16	64.86±2.03
		+ LAGER$_{nearest}$	59.13±3.96	70.83±3.18	64.33±2.45	**60.87±2.04**	70.38±1.86	**65.27±1.79**
		+ LAGER$_{angles}$	**61.11±3.53**	**72.16±2.74**	**66.13±2.76**	60.84±1.5	70.31±1.76	65.23±1.46
	8	Vanilla	61.69±2.93	69.9±3.3	65.51±2.75	62.98±0.94	72.07±1.24	67.18±1.07
		+ LAGER$_{nearest}$	**63.31±2.0**	71.99±1.87	**67.35±1.46**	**63.07±2.31**	72.08±2.47	67.27±2.34
		+ LAGER$_{angles}$	61.92±4.21	**72.4±2.85**	66.63±2.5	63.28±1.75	**72.23±2.0**	**67.46±1.84**
	9	Vanilla	62.54±2.22	71.17±3.65	66.55±2.54	63.93±0.5	72.63±0.63	67.96±0.15
		+ LAGER$_{nearest}$	62.62±2.47	71.26±2.1	66.64±1.95	**64.19±2.53**	**72.99±1.21**	**68.3±1.88**
		+ LAGER$_{angles}$	**63.53±3.81**	**71.94±2.64**	**67.44±3.08**	63.72±1.75	72.79±1.29	67.94±1.4
	10	Vanilla	63.49±2.7	72.17±2.21	67.59±2.33	66.18±0.99	73.79±0.8	69.92±0.76
		+ LAGER$_{nearest}$	**63.9±2.87**	71.89±3.03	**67.63±2.62**	66.09±1.34	74.0±0.5	69.82±0.9
		+ LAGER$_{angles}$	62.77±3.98	**72.71±1.67**	67.33±2.78	**66.37±1.24**	**74.19±0.69**	**70.06±0.95**

Table 7: Evaluation results with LayoutLMv2 as baseline on all few-shot sizes. **Bold** indicates best model

	$\mathcal{P}$	Model	FUNSD Precision	FUNSD Recall	FUNSD F-1	CORD Precision	CORD Recall	CORD F-1
LayoutLMv3	1	Vanilla	22.93±6.21	39.7±5.94	28.85±6.35	36.56±5.46	47.58±6.44	41.33±5.88
		+ LAGER$_{nearest}$	**28.74±8.4**	**35.99±7.71**	**31.49±7.63**	**38.87±6.57**	**49.83±6.76**	**43.64±6.73**
		+ LAGER$_{angles}$	26.11±6.69	32.53±6.29	28.23±5.24	37.52±6.34	48.7±6.6	42.36±6.54
	2	Vanilla	44.29±6.14	58.96±7.2	50.43±6.03	47.21±6.25	58.99±4.94	52.41±5.85
		+ LAGER$_{nearest}$	**49.82±6.06**	**59.55±8.91**	**54.09±6.54**	**48.68±5.72**	60.19±4.23	**53.79±5.24**
		+ LAGER$_{angles}$	46.8±6.46	58.15±8.92	51.61±6.42	48.15±5.07	**60.3±3.57**	53.51±4.58
	3	Vanilla	59.66±4.92	72.2±7.65	65.29±5.92	51.34±6.55	62.49±5.47	56.34±6.2
		+ LAGER$_{nearest}$	**62.18±5.13**	**73.12±7.3**	**67.12±5.56**	**53.08±7.32**	**64.3±5.55**	**58.1±6.73**
		+ LAGER$_{angles}$	60.73±5.09	72.41±7.64	65.97±5.71	52.77±7.17	63.72±5.55	57.68±6.63
	4	Vanilla	65.32±3.89	77.97±2.26	71.06±3.04	54.18±5.01	64.92±3.76	59.04±4.53
		+ LAGER$_{nearest}$	**67.86±3.3**	**78.73±2.57**	**72.86±2.69**	**56.28±4.24**	**66.47±3.29**	**60.94±3.86**
		+ LAGER$_{angles}$	65.93±3.28	77.22±3.45	71.08±2.81	55.38±4.63	65.99±3.79	60.21±4.3
	5	Vanilla	67.14±5.17	77.88±2.62	72.07±4.01	58.55±2.82	67.03±2.46	62.49±2.57
		+ LAGER$_{nearest}$	69.6±2.57	79.72±1.66	74.3±1.94	**59.84±3.27**	68.36±2.34	**63.8±2.76**
		+ LAGER$_{angles}$	**70.32±1.41**	**80.86±1.23**	**75.22±1.1**	59.37±4.09	**68.48±3.08**	63.58±3.63
	6	Vanilla	71.19±3.75	80.83±1.09	75.68±2.58	60.91±3.51	69.16±2.76	64.76±3.16
		+ LAGER$_{nearest}$	**72.71±3.42**	81.53±1.98	**76.84±2.58**	**61.8±5.14**	70.0±3.75	65.63±4.53
		+ LAGER$_{angles}$	72.31±3.7	**81.65±1.81**	76.67±2.7	61.56±4.49	**70.3±3.33**	**65.63±3.98**
	7	Vanilla	72.44±3.56	81.56±1.12	76.68±1.95	62.26±3.78	70.3±3.02	66.03±3.43
		+ LAGER$_{nearest}$	74.48±2.42	82.4±1.1	78.22±1.49	**62.84±4.17**	**71.05±3.26**	**66.68±3.77**
		+ LAGER$_{angles}$	**74.63±2.69**	**83.18±2.04**	**78.65±2.06**	62.77±4.17	71.01±3.15	66.62±3.72
	8	Vanilla	74.31±2.19	81.75±2.6	77.85±2.29	64.49±3.24	72.21±2.17	68.12±2.77
		+ LAGER$_{nearest}$	76.27±1.44	83.41±1.73	79.66±1.14	64.89±4.38	72.22±3.19	68.35±3.84
		+ LAGER$_{angles}$	**76.41±2.18**	**83.98±1.49**	**79.99±1.24**	**65.1±4.14**	**72.27±2.87**	**68.49±3.56**
	9	Vanilla	75.9±1.53	82.52±1.36	79.06±1.24	65.89±2.8	73.44±1.87	69.45±2.39
		+ LAGER$_{nearest}$	76.83±1.95	83.23±1.61	79.89±1.51	**66.84±3.25**	**73.62±1.85**	**70.05±2.6**
		+ LAGER$_{angles}$	**76.95±2.03**	**84.43±1.65**	**80.5±1.59**	66.73±3.99	73.45±2.35	69.91±3.25
	10	Vanilla	76.1±2.31	82.65±2.34	79.24±2.27	66.72±2.65	73.58±2.08	69.97±2.34
		+ LAGER$_{nearest}$	**77.44±2.65**	**84.43±1.83**	80.77±2.02	**67.68±3.49**	73.64±2.57	**70.53±3.05**
		+ LAGER$_{angles}$	77.39±2.53	84.57±1.53	**80.81±1.79**	67.05±3.2	**73.64±1.93**	70.18±2.61

Table 8: Evaluation results with LayoutLMv3 as baseline on all few-shot sizes. **Bold** indicates best model