A Graph Construction Procedure

The proposed Doc-GCN contains graphs covering four different aspects of properties of PDF pages, including syntactic, semantic, density, and appearance/visual information, to harmonize and integrate heterogeneous aspects for Document Layout Analysis. In this section, we would like to describe the graph construction procedure for each aspect graph. The syntactic and semantic graphs share the same graph structure, while the density and appearance graphs use the other graph structures.

A.1 Density-aspect and Appearance-aspect Graph construction

The graph construction scenarios for both density-aspect and appearance-aspect graphs can be found in Figure 1. We demonstrate the detailed steps as follows:

1. As is illustrated in step 1, given a scanned document page, we identify each document layout component as a segment based on the bounding box provided in the datasets.

2. In step 2, the bounding box information for each segment is defined by the left-top and right-bottom coordinates.

3. To construct the Density-aspect/Appearance-aspect graphs in step 3, we treat each segment as a graph node and connect each segment node with its two closest neighbors based on the gap between their bounding boxes. These two closest neighbors could be both vertically and horizontally located (if a two-column page is applied). We initialize the $\mathcal{G}_{den1}$ node values as the ratio of density following the Equation 1.

$$\text{Ratio}_{density} = \frac{\text{\#tokens}}{\text{Area size of bbox}} \quad (1)$$

   $\mathcal{G}_{den2}$ node values are initialized as the total number of characters in each segment.

   For appearance graph, $\mathcal{G}_{appr}$, the node values are initialized by the visual features of each segment. Such visual features are extracted from the pretrained ResNet-101.

A.2 Syntactic-aspect and Semantic-aspect Graph Construction

The graph construction scenarios for both syntactic-aspect and semantic-aspect graphs can be found in Figure 2a. We demonstrate the detailed steps as follows:

1. As is illustrated by step 1, given a scanned document page, we identify each document layout component as a segment based on the bounding box provided in the dataset.

2. In step 2, we represent the hierarchical parent-child relations between segments. For the training and validation set, we use the parent-child relations that are directly derived from the source files provided in the datasets: we first apply OCR detection to each segment in step 1. Based on the detected OCR text tokens, we then use fuzzy string matching to map each segment with the corresponding element in the *XML/* LaTeX source files for PubLayNet/DocBank and identify the parent-child relations based on the hierarchical structure embodied in these source files; FUNSD provides the parent-child relations directly in the $.json$ annotation files. We then train a transformer-based relation prediction model on the (training + validation) set utilizing those extracted parent-child relations and predict the parent-child relations for the test set. The details of the transformer-based relation prediction model are provided in Section A.3.

3. To construct the Syntactic and Semantic graph, as is demonstrated in step 3, we treat each segment as a node of the graph. We then connect the parent-child relations nodes (derived from step 2) and set the edge value as 1. Regarding the node representation, we use the first-level and second-level syntactic parsing results as the node information for the Syntactic Graph $\mathcal{G}_{syn1}$ and $\mathcal{G}_{syn2}$ respectively before the node encoding. For the Semantic Graph G_{semc}, we pass the OCR tokens within each segment node to the pretrained BERT and extract the $[CLS]$ as the initialized node embedding.

A.3 Relation Prediction Model

As described in Section A.2, we apply a relation prediction model to predict the parent-child relations between segments when constructing both syntactic-aspect and semantic-aspect graphs for data in the test set.

Given a PDF page with multiple segments, we first encode each segment's textual and visual features, as illustrated on the left side in Figure 2b.

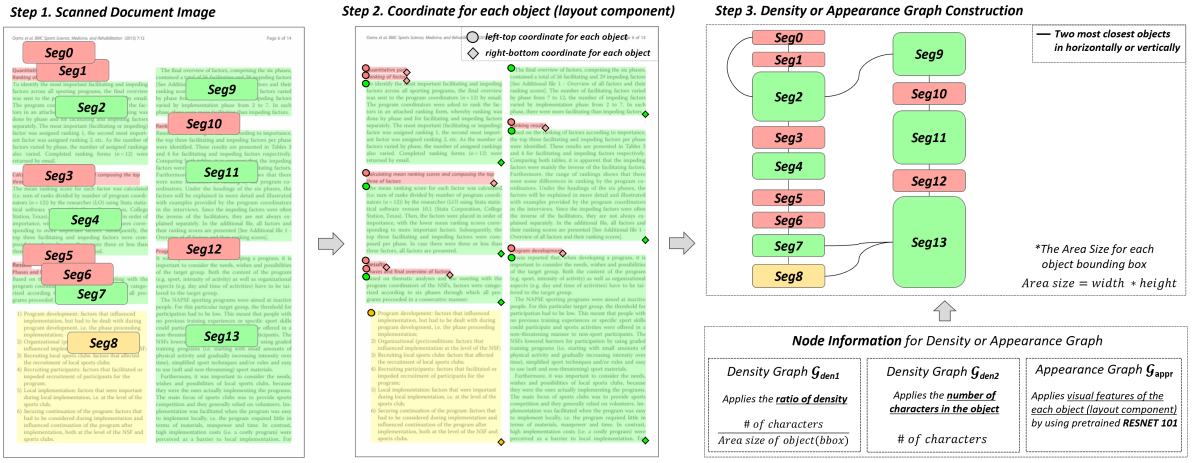


Figure 1: Density Graph and Appearance Graph Construction

	PubLayNet	FUNSD	DocBank
Bert-L	97.06	80.86	86.42
RoBERTa-L	96.15	79.47	86.67
LayoutLM-L	96.94	78.90	83.21
Doc-GCN-L	**98.22**	**85.40**	**90.07**

Table 1: Performances comparison (F1 score%) based on large (L) pretrained models.

On the one hand, we feed the OCR token sequence $w_1, w_2, ..., w_n$ (derived from step 2 in Section A.2) of the segment into the pretrained BERT model and adopt the generated representation of the $[CLS]$ token as the textual embedding $h_{[CLS]}$. On the other hand, the cropped image of each segment (based on the bounding box) is fed into the pretrained ResNet-101 model, from which the output feature of Res5 layer is used as the visual embedding V_{Res5}. We then concatenate the textual embedding $h_{[CLS]}$ and visual embedding as the textual-visual embedding for each segment.

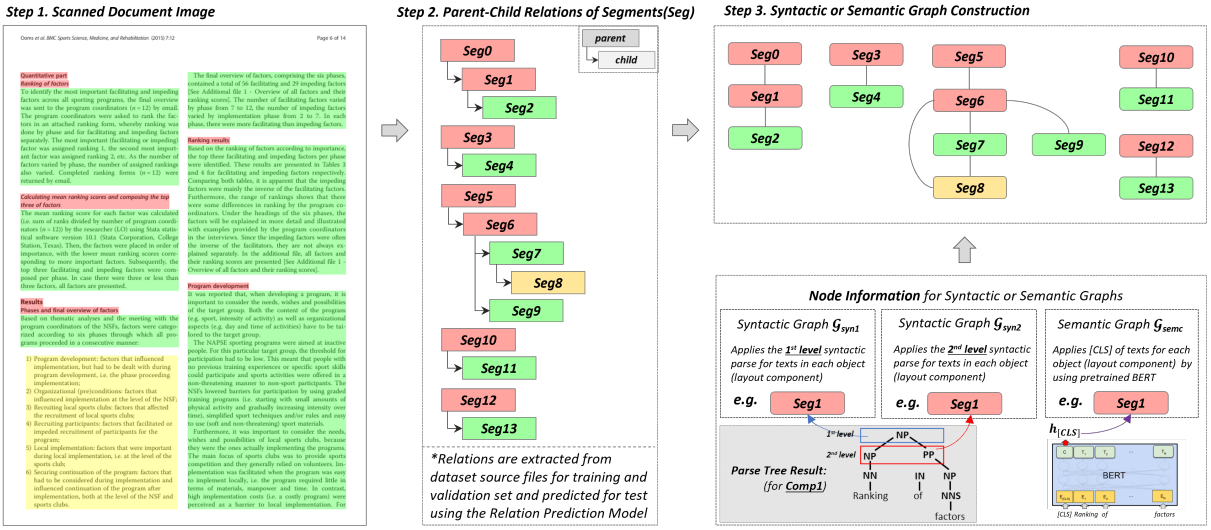
The relation prediction model is shown on the right side in Figure 2b. We represent the sequence of segments in a PDF page using their textual-visual embedding and sum it up with the positional embedding as the input embedding, which is then passed to a single-layer transformer encoder to derive the contextualized representation for the segments. We then apply two linear transformations, Q-Transform and K-Transform, to generate the $query$ and key representation. An attention score matrix is generated from the softmax-normalized dot-product results of the two representations, based on which the parent segment is selected from the segments in the input sequence. To handle those segments without parents, we append a fake segment at the end of the input sequence initialized with all zeros as a place holder, which is treated as the target parent for the non-parent segment.

B Additional Case Studies

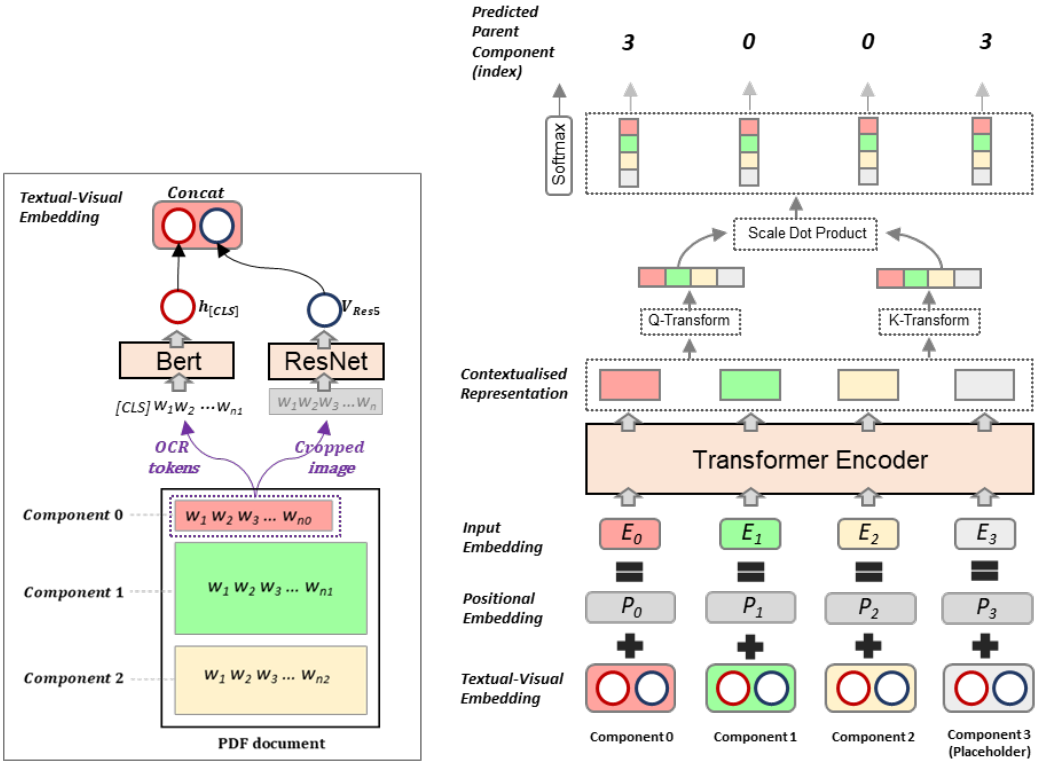
We provided one additional quality result on the document of PubLayNet (Figure 3), FUNSD (Figure 4,5) and DocBank (Figure 6,7),respectively. As shown in Figure 3, both RoBERTa-base and Faster-RCNN incorrectly recognize the *List* into *Text*, whereas our Doc-GCN successfully recognize all segments for Publaynet Dataset.

As shown in Figure 4, Doc-GCN also shows its remarkable performance by correctly recognising the top-right component highlighted by the green tick in Figure 4d while the other two baseline models wrongly classified this component into *Answer*. Additionally, ours detected *Other* components such as page number, company name, and date time correctly. This is another highlight of the great potential of our multi-modal aspects, even with the small-sized of the training dataset. Similar trend also can be found in Figure 5.

Figure 6, a dataset with the scientific academic papers, shows that Doc-GCN produced all components correctly. However, BERT could not detect *Date* (published date) and *Abstract*, which are the most important components of the publication. RoBERTa also incorrectly recognised the *Date* (published date) and *Author*. Thus it is clear to see that our Doc-GCN consistently works well in detecting paragraphs and equations. As well as, in Figure 7, only our Doc-GCN can get the exactly same results as the ground truth.



(a) Semantic Graph and Syntactic Graph Construction

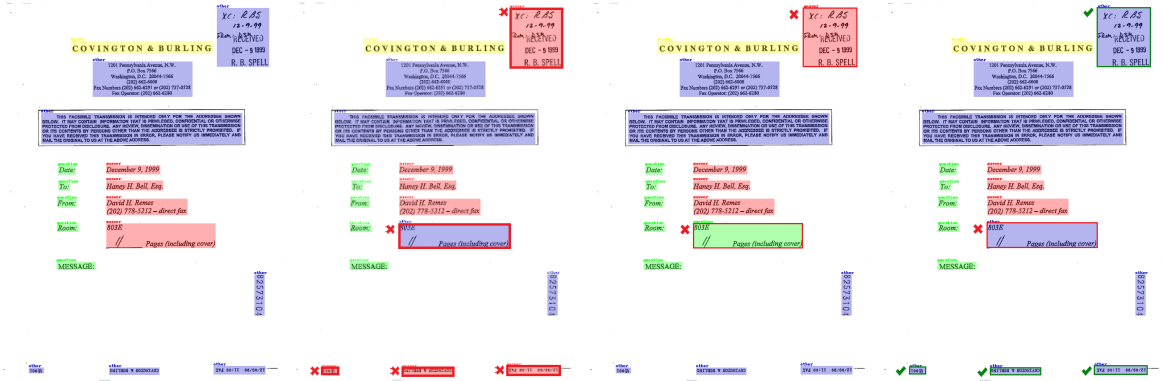


(b) Relation Prediction Model

Figure 2: Semantic Graph and Syntactic Graph Construction flow with Parent-Child Relation prediction model architecture.

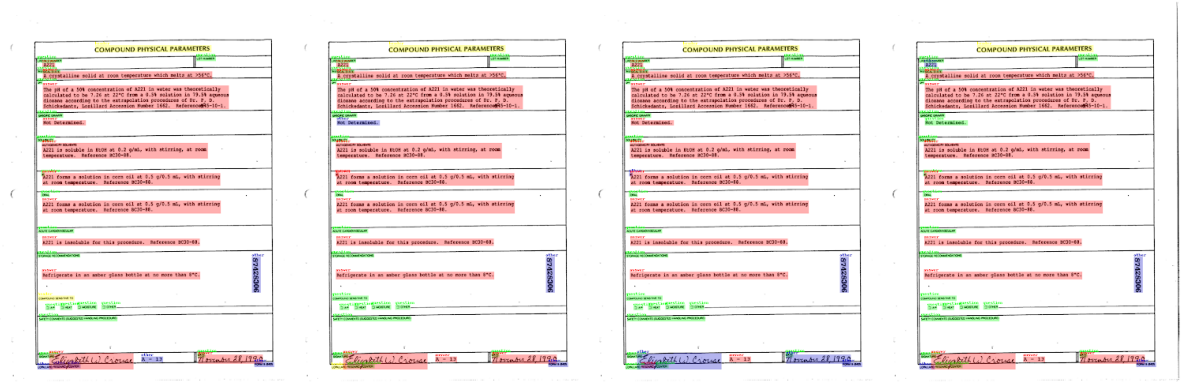
(a) **Ground Truth** (b) **RoBERTa-base** (c) **Faster-RCNN** (d) **Doc-GCN**

Figure 3: Example output of Top3 models for a **PubLayNet** page. The color of layout component labels are: Text, Title, List, Table, Figure. Our Doc-GCN classified all layout components accurately.



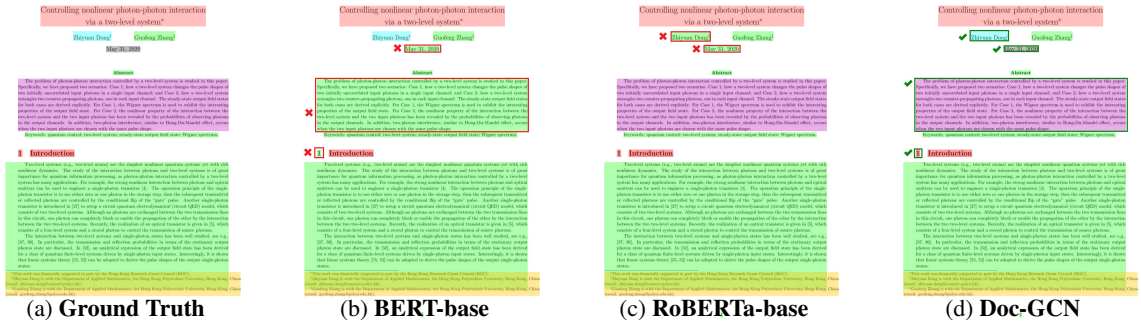
(a) **Ground Truth** (b) **BERT-base** (c) **RoBERTa-base** (d) **Doc-GCN**

Figure 4: Visualisation of Top3 models for a FUNSD page. The color of layout components are: Question, Answer, Header, Other. Our Doc-GCN produced the best recognition results, especially for the *Other* components.



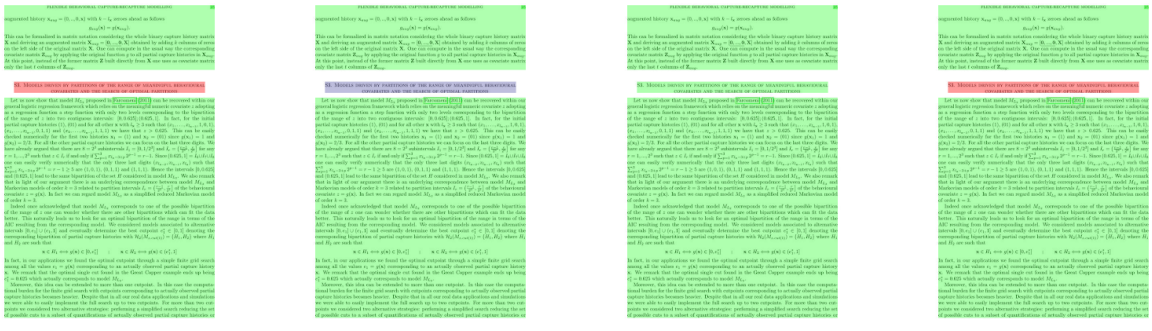
(a) **Ground Truth** (b) **BERT-base** (c) **RoBERTa-base** (d) **Doc-GCN**

Figure 5: Visualisation of Top3 models for a **FUNSD** page. The color of layout components are: Question, Answer, Header, Other. Our Doc-GCN produced the best recognition results, especially for the *Other* components.



(a) **Ground Truth** (b) **BERT-base** (c) **RoBERTa-base** (d) **Doc-GCN**

Figure 6: Visualization of Top3 models (BERT-base, RoBERTa-base and Doc-GCN) for a DocBank PDF page. The color of layout component labels are: Abstract, Author, Caption, Date, Equation, Figure, Footer, List, Paragraph, Reference, Section, Table, Title. Our Doc-GCN classified all layout components accurately.



(a) **Ground Truth** (b) **BERT-base** (c) **RoBERTa-base** (d) **Doc-GCN**

Figure 7: Visualization of Top3 models for a **DocBank** PDF page. The color of layout component labels are: Abstract, Author, Caption, Date, Equation, Figure, Footer, List, Paragraph, Reference, Section, Table, Title. Our Doc-GCN classified all layout components accurately.