

Doc2Graph: a Task Agnostic Document Understanding Framework based on Graph Neural Networks

Andrea Gemelli¹, Sanket Biswas², Enrico Civitelli¹, Josep Lladós², and Simone Marinai¹

¹ Dipartimento di Ingegneria dell'Informazione (DINFO)
Università degli studi di Firenze, Italy

{andrea.gemelli, enrico.civitelli, simone.marinai}@unifi.it

² Computer Vision Center & Computer Science Department
Universitat Autònoma de Barcelona, Spain
{sbiswas, josep}@cvc.uab.es

Abstract. Geometric Deep Learning has recently attracted significant interest in a wide range of machine learning fields, including document analysis. The application of Graph Neural Networks (GNNs) has become crucial in various document-related tasks since they can unravel important structural patterns, fundamental in key information extraction processes. Previous works in the literature propose task-driven models and do not take into account the full power of graphs. We propose Doc2Graph, a task-agnostic document understanding framework based on a GNN model, to solve different tasks given different types of documents. We evaluated our approach on two challenging datasets for key information extraction in form understanding, invoice layout analysis and table detection. Our code is freely accessible on <https://github.com/andreagemelli/doc2graph>.

Keywords: Document Analysis and Recognition · Graph Neural Networks · Document Understanding · Key Information Extraction · Table Detection

1 Introduction

Document Intelligence deals with the ability to read, understand and interpret documents. Document understanding can be backed by graph representations, that robustly represent objects and relations. Graph reasoning for document parsing involves manipulating structured representations of semantically meaningful document objects (titles, tables, figures) and relations, using compositional rules. Customarily, graphs have been selected as an adequate framework for leveraging structural information from documents, due to their inherent representational power to codify the object components (or semantic entities) and their pairwise relationships. In this context, recently graph neural networks (GNNs)

have emerged as a powerful tool to tackle the problems of Key Information Extraction (KIE) [6,35], Document Layout Analysis (DLA) which includes well-studied sub-tasks like table detection [25,26], table structure recognition [20,34] and table extraction [9], Visual Question Answering (VQA) [18,17], synthetic document generation [4] and so on.

Simultaneously, the common state-of-the-art practice in the document understanding community is to utilize the power of huge pre-trained vision-language models [1,32,33] that learn whether the visual, textual and layout cues of the document are correlated. Despite achieving superior performance on most document understanding tasks, large-scale document pre-training comes with a high computational cost both in terms of memory and training time. We present a solution that does not rely on huge vision-language model pre-training modules, but rather recognizes the semantic text entities and their relationships from documents exploiting graphs. The solution has experimented on two challenging benchmarks for forms [15] and invoices [10] with a very small amount of labeled training data.

Inspired by some prior works [8,25,26], we introduce *Doc2Graph*, a novel task-agnostic framework to exploit graph-based representations for document understanding. The proposed model is validated in three different challenges, namely KIE in form understanding, invoice layout analysis and table detection. A graph representation module is proposed to organize the document objects. The graph nodes represent words or the semantic entities while edges the pairwise relationships between them. Finding the optimal set of edges to create the graph is anything but trivial: usually in literature heuristics are applied, e.g. using a visibility graph [25]. In this work, we do not make any assumption a priori on the connectivity: rather we attempt to build a fully connected graph representation over documents and let the network learn by itself what is relevant.

In summary, the primary contributions of this work can be summarized as follows:

- Doc2Graph, the first task-agnostic GNN-based document understanding framework, evaluated on two challenging benchmarks (form and invoice understanding) for three significant tasks, without any requirement of huge pre-training data;
- We propose a general graph representation module for documents, that do not rely on heuristics to build pairwise relationships between words or entities;
- A novel GNN architectural pipeline with node and edge aggregation functions suited for documents, that exploits the relative positioning of document objects through polar coordinates.

The rest of the paper is organized as follows. In section 2 we review the state-of-the-art in graph representation learning and vision-language models for document understanding. Section 3 provides the details of the main methodological contribution. The experimental evaluation is reported in section 4. Finally, the conclusions are drawn in section 5.

2 Related Work

Document understanding has been studied extensively in the last few years, owing to the advent of deep learning, but has been reformulated in a recent survey by Borchmann et. al. [5]. The tasks range from KIE performed for understanding forms [15], receipts [14] and invoices [10], to multimodal comprehension of both visual and textual cues in a document for classification [32,33]. It also includes the DLA task where recent works focus on building an end-to-end framework for both detection and classification of page regions [3,2]. Table detection [25,26], structure recognition [20,24] and extraction [9,30] in DLA gathered some special attention in recent years due to the high variability of layouts that make the both necessary to be solved and challenging to be tackled. In addition, question answering [21,29] has emerged as an extension of the KIE task principle, where a natural language question replaces a property name. Current state-of-the-art approaches [1,13,22,32,33] on these document understanding tasks have utilized the power of large pre-trained language models, relying on language more than the visual and geometrical information in a document and also end up using hundreds of millions of parameters in the process. Moreover, most of these models are trained with a huge transformer pipeline, which requires an immense amount of data during pre-training. In this regard, Davis et al. [7] and Sarkar et al. [28] proposed language-agnostic models. In [7] they focused on the entity relationship detection problem in forms [15] using a simple CNN as a text line detector and then detecting key-value relationship pairs using a heuristic based on each relationship candidate score generated from the model. Sarkar et al. [28] rather focused on extracting the form structure by reformulating the problem as a semantic segmentation (pixel labeling) task. They used a U-Net based architectural pipeline, predicting all levels of the document hierarchy in parallel, making it quite efficient.

GNN for document understanding was first introduced for mainly key DLA sub-tasks that include table detection [25] and table structure recognition [23]. The key idea behind its introduction was to utilize the powerful geometrical characteristics of a document using GNN and then to preserve the privacy of confidential textual content (especially for administrative documents) during training, making the model language-independent and more structure-reliant as proposed in [25] for detection of tables in invoices. Carbonell et. al. [6] used graph convolutional networks (GCNs) to solve the entity(word) grouping, labeling and entity linking tasks for form analysis. They used the information of the bounding boxes and word embeddings as the principal node features and do not include any visual features, while they used k-nearest neighbours (KNNs) to encode the edge information. The FUDGE [8] framework was then developed for form understanding as an extension of [7] to greatly improve the state-of-the-art on both the semantic entity labeling and entity linking tasks by proposing relationship pairs using the same detection CNN as in [7]. Then a graph convolutional network (GCN) was deployed with plugged visual features from the CNN so that semantic labels for the text entities were predicted jointly with the key-value relationship pairs, as they are quite related tasks.

Inspired by this influential prior work [8], we aim to propose a task-agnostic GNN-based framework called *Doc2Graph* that adapts a similar joint prediction of both the tasks, semantic entity labeling and entity linking using a node classification and edge classification module respectively. Doc2Graph is established to tackle multiple challenges ranging from KIE for form understanding to layout analysis and table detection for invoice understanding, without needing any kind of huge data pre-training and being lightweight and efficient.

3 Method

In this section, we present the proposed approach. First, we describe the pre-processing step that converts document images into graphs. Then, we describe the GNN model designed to tackle different kinds of tasks.

3.1 Documents graph structure

A graph is a structure made of nodes and edges. A graph can be seen as a language model representing a document in terms of its segments (text units) and relationships. A preprocessing step is required. Depending on the task, different levels of granularity have to be considered for defining the constituent objects of a document. They can be single words or entities, that is, groups of words that share a certain property (e.g., the name of a company). In our work we try both as the starting point of the pipeline: we apply an OCR to recognize words, while a pre-trained object detection model for detecting entities. The chosen objects, once found, constitute the nodes of the graph.

At this point, nodes need to be connected through edges. Finding the optimal set of edges to create the graph is anything but trivial: usually in literature heuristics are applied, e.g. using a visibility graph [25]. These approaches: (i) do not generalise well on different layouts; (ii) strongly rely on the previous node detection processes, which are often prone to errors; (iii) generate noise in the connections, since bounding box of objects could cut out important relations or allow unwanted ones; (iv) exclude in advance sets of solutions, e.g. answers far from questions. To avoid those behaviours, we do not make any assumption a priori on the connectivity: we build a fully connected graph and we let the network learn by itself what relations are relevant.

3.2 Node and edge features

In order to learn, suitable features should be associated to nodes and edges of the graph. In documents, this information can be extracted from sources of different modalities, such as visual, language and layout ones. Different methods can be applied to encode a node (either word or entity) to enrich its representation. In our pipeline, with the aim to possibly keep it lightweight, we include:

- a language model to encode the text. We use the spaCy large English model to get word vector representations of words and entities;

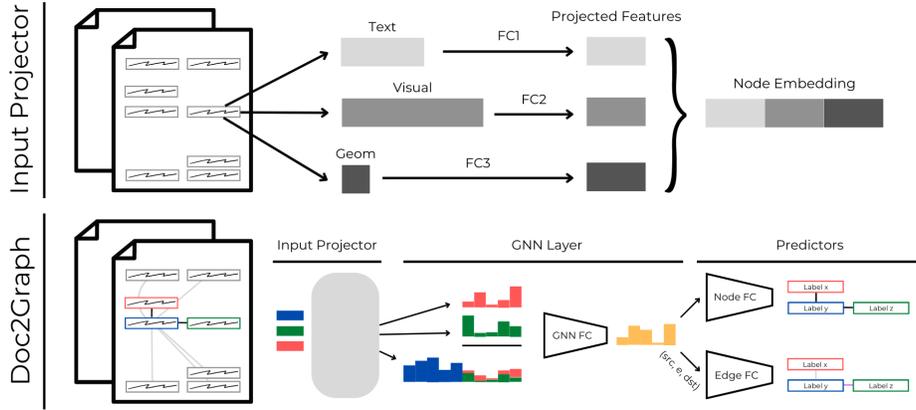


Fig. 1: **Our proposed Doc2Graph framework.** For visualisation purposes, the architecture shows the perspective of one node (the blue one in Doc2Graph).

- a visual encoder to represent style and formatting. We pretrain a U-Net[27] on FUNSD for entities segmentation. Since U-Net uses feature maps at different encoder’s layers to segment the images, we decide to use all these information as visual features. Moreover, it is important to highlight that, for each features map, we used a RoI Alignment layer to extract the features relative to each entities bounding box;
- the absolute normalized positions of objects inside a document; layout and structure are meaningful features to include in industrial documents, e.g. for key-value associations.

As for the edges, to the best of our knowledge, we propose two new sets of features to help both the node and the edge classification tasks:

- a normalized euclidean distance between nodes, by means of the minimum distance between bounding boxes. Since we are using a fully connected graph this is crucial for the aggregation node function in use to keep locality property during the message passing algorithm;
- relative positioning of nodes using polar coordinates. Each source node is considered to be in the center of a Cartesian plane and all its neighbors are encoded by means of distance and angle. We discretize the space into bins (one-hot encoded), which number can be chosen, instead of using normalized angles: a continuous representation of the angle is challenging because, for instance, two points at the same distance with angles 360° and 0° would be encoded differently.

3.3 Architecture

Each node feature vector passes through our proposed architecture (Fig. 1, visualization of GNN layer inspired by “A Gentle Introduction to GNNs”): the

connectivity defines the neighborhood for the message passing, while the weight learnable matrices are shared across all nodes and edges, respectively. We make use of four different components:

- Input Projector: this module applies as many fully connected (FC) layers as there are different modalities in use, to project each of their representations into inner spaces of the same dimension; e.g., we found it to be not very informative combine low dimensional geometrical features with high dimensional visual ones, as they are;
- GNN Layer: we make use of a slightly different version of GraphSAGE [11]. Using a fully connected graph, we redefine the aggregation strategy (eq. 2);
- Node Predictor: this is a FC layer, that maps the representation of each node into the number of target classes;
- Edge Predictor: this is a two-FC layer, that assigns a label to each edge. To do so, we propose a novel aggregation on edges (eq. 3).

GNN Layer Our version of GraphSAGE slightly differs in the neighborhood aggregation. At layer l given a node i , h_i its inner representation and $N(i)$ its set of neighbors, the aggregation is defined as:

$$h_{N(i)}^{l+1} = \text{aggregate}(\{h_j^l, \forall j \in N(i)\}) \quad (1)$$

where *aggregate* can be any permutation invariant operation, e.g. sum or mean. Usually, in other domains, the graph structure is naturally given by the data itself but, as already stated, in documents this can be challenging (sec. 3.1). Then, given a document, we redefine the above equation as:

$$h_{N(i)}^{l+1} = \frac{c}{|\mathcal{Y}(i)|} \sum_{j \in \mathcal{Y}(i)} h_j^l \quad (2)$$

where $\mathcal{Y}(i) = \{j \in N(i) : |i - j| < \text{threshold}\}$, $|i - j|$ is the Euclidean distance of nodes i and j saved (normalized between 0 and 1) on their connecting edge, and c is a constant scale factor.

Edge Predictor We consider each edge as a triplet (src, e, dst) : e is the edge connecting the source (src) and destination (dst) node. The edge representation h_e to feed into the two-FC classifier is defined as:

$$h_e = h_{src} \parallel h_{dst} \parallel cls_{src} \parallel cls_{dst} \parallel e_{polar} \quad (3)$$

where h_{src} and h_{dst} are the node embeddings output of the last GNN layer, cls_{src} and cls_{dst} are the softmax of the output logits of the previous node predictor layer, e_{polar} are the polar coordinates described in sec 3.2 and \parallel is the concatenation operator. These choices have been made because: (i) relative positioning on edges is stronger compared to absolute positioning on nodes: the local property introduced by means of polar coordinates can be extended to different

data, e.g. documents of different sizes or orientations; (ii) if the considered task comprise also the classification of nodes, their classes may help in the classification of edges, e.g. in forms it should not possible to find an answer connected to another answer.

Given the task, graphs can be either undirected or directed: both are represented with two or one directed edge between nodes, respectively. In the first case, the order does not matter and so the above formula can be redefined as:

$$h_e = (h_{src} + h_{dst}) \parallel cls_{src} \parallel cls_{dst} \parallel e_{polar} \quad (4)$$

4 Experiments and Results

In this chapter we present experiments of our method on two different datasets, FUNSD and RVL-CDIP invoices, to tackle three tasks: entity linking, layout analysis and table detection. We also discuss results compared to other methods.

4.1 Proposed model

We performed ablation studies on our proposed model for entity linking on FUNSD without contribution and classification of nodes (Fig. 1), since we found it to be the most challenging task. In Tab. 1 we report different combinations of features and hyperparameters. Geometrical and textual features make the largest contribution, while visual features bring almost three points more to the Key-Value F1 score by an important increase in terms of network parameters (2.3 times more). Textual and geometrical features remain crucial for the task at hand, and their combination increase by a large amount both of their scores when used in isolation. This may be due to two facts: (i) our U-Net has not been included during the GNN training time (as done in [8]), unable to adjust the representation for spotting key-value relationship pairs; (ii) the segmentation task used to train the backbone do not yield useful features for that goal (as shown in Tab. 1). The hyperparameters shown in the table refer to the edge predictor (EP) inner layer input dimension and the input projector fully connected (IP FC) layers (per each modality) output dimension, respectively. A larger EP is much more informative for the classification of links into ‘none’ (cut edges, meaning no relationship) or ‘key-value’, while more dimensions for the projected modalities helped the model to better learn the importance of their contributions. These changes bring an improvement of 13 points on the key-value F1 scores, between the third and fourth line of the table where we keep the features fixed. We do not report the score relative to others network settings since their changes only brought a decrease overall metrics. We use a learning rate of 10^{-3} and a weight decay of 10^{-4} , with a dropout of 0.2 over the last FC layer. The threshold over neighbor nodes and their contribution scale factor (sec. 3.3) are fixed to 0.9 and 0.1, respectively. The bins to discretize the space for angles (sec. 3.3) are 8. We apply one GNN layer before the node and edge predictors.

Features			F_1 per classes (†)					
Geometric	Text	Visual	EP Inner dim	IP FC dim	None	Key-Value	AUC-PR (†)	# Params $\times 10^6$ (‡)
✓	✗	✗	20	100	0.9587	0.1507	0.6301	0.025
✗	✓	✗	20	100	0.9893	0.1981	0.5605	0.054
✓	✓	✗	20	100	0.9941	0.4305	0.7002	0.120
✓	✓	✗	300	300	0.9961	0.5606	0.7733	1.18
✓	✓	✓	300	300	0.9964	0.5895	0.7903	2.68

Table 1: **Ablation studies of Doc2Graph model.** EP Inner dim and IP FC dim show edge predictor layer input dimension and the input projector fully connected layers output dimension, respectively. AUC-PR refers to the key-value edge class. The # Params refers to Doc2Graph trainable parameters solely.

4.2 FUNSD

Dataset The dataset [15] comprises 199 real, fully annotated, scanned forms. The documents are selected as a subset of the larger RVL-CDIP [12] dataset, a collection of 400,000 grayscale images of various documents. The authors define the Form Understanding (FoUn) challenge into three different tasks: word grouping, semantic entity labeling and entity linking. A recent work [31] found some inconsistency in the original labeling, which impeded its applicability to the key-value extraction problem. In this work, we are using the revised version of FUNSD.

Entity Detection Our focus is on the GNN performances but, for comparison reasons, we used a YOLOv5 small [16] to detect entities (pretrained on COCO [19]). In [15] the word grouping task is evaluated using the ARI metric: since we are not using words, we evaluated the entity detection with F1 score using two different IoU thresholds (Tab. 2). For the semantic entity labeling and entity

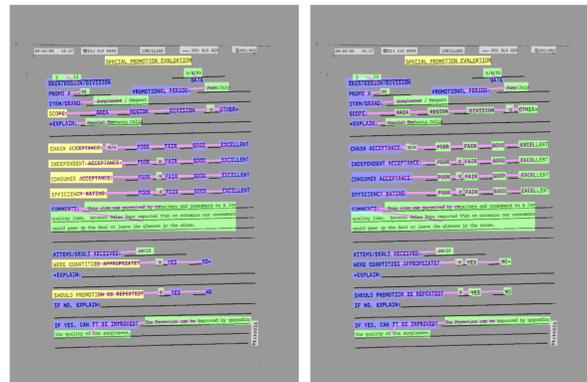


Fig. 2: Image taken from [31]: the document on the right is the revised version of the document on the left, where some answers (green) are mislabeled as question (blue), and some questions (blue) are mislabeled as headers (yellow)

IoU	Metrics (\uparrow)			% Drop Rate (\downarrow)	
	Precision	Recall	F ₁	Entity	Link
0.25	0.8728	0.8712	0.8720	12.72	16.63
0.50	0.8132	0.8109	0.8121	18.67	25.93

Table 2: **Entity detection results.** YOLOv5 [16]-small performances on the entity detection task.

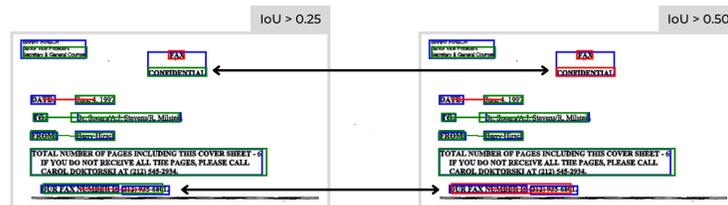


Fig. 3: Blue boxes are FUNSD entities ground truth, green boxes are the correct detected one (with $\text{IoU} > 0.25/0.50$), while red boxes are the false positive ones.

linking tasks we use $\text{IoU} > 0.50$ as done in [8]: we did not perform any optimization on the detector model, which introduces a high drop rate for both entities and links. We create the graphs on top of YOLO detections, linking the ground truth accordingly (Fig. 3): false positive entities (red boxes) are labeled as class ‘other’, while false negative entities cause some key-value pairs to be lost (red links). The new connections created as a consequence of wrong detections are considered false positives and labeled as ‘none’.

Numerical results We trained our architecture (sec. 3.3) with a 10-fold cross validation. Since we found high variance in the results, we report both mean

Method	GNN	F_1 (\uparrow)			# Params $\times 10^6$ (\downarrow)
		Semantic Entity Labeling	Entity Linking		
BROS [13]	\times	0.8121	0.6696	138	
LayoutLM [33,13]	\times	0.7895	0.4281	343	
FUNSD [15]	\checkmark	0.5700	0.0400	-	
Carbonell et al. [6]	\checkmark	0.6400	0.3900	201	
FUDGE w/o GCN [8]	\times	0.6507	0.5241	12	
FUDGE [8]	\checkmark	0.6652	0.5662	17	
Doc2Graph + YOLO	\checkmark	0.6581 ± 0.006	0.3882 ± 0.028	13.5	
Doc2Graph + GT	\checkmark	<u>0.8225 ± 0.005</u>	0.5336 ± 0.036	6.2	

Table 3: **Results on FUNSD.** The results have been shown for both semantic entity labeling and entity linking tasks with their corresponding metrics.

and variance over the 10 best models chosen over their respective validation sets. The objective function in use (L) is based on both node (L_n) and edge (L_e) classification tasks: $L = L_n + L_e$. In Tab. 3 we report the performances of our model Doc2Graph compared to other language models [13,33] and graph-based techniques [6,8]. The number of parameters # Params refer to the trainable Doc2Graph pipeline (that includes the U-Net and YOLO backbones); for the spaCy word-embedding details, refer to [their documentation](#). Using YOLO our network outperforms [6] for semantic entity labeling and meets their model on entity linking, using just 13.5 parameters. We could not do better than FUDGE, which still outperforms our scores. Their backbone is trained for both tasks along with the GCN (GCN that adds just minor improvements). The gap, especially on entity linking, is mainly due to the low contributions given by our visual features (Tab. 1) and the detector in use (Tab. 3). We also report the results of our model initialized with ground truth (GT) entities, to show how it would perform in the best case scenario. Entity linking remains a harder task compared to semantic entity labeling and only complex language models seem to be able to solve it. Moreover, for the sake of completeness, we highlight that, with good entity representations, our model outperforms all the considered architectures for the Semantic Entity Labeling task. Finally, we want to further stress that the main contribution of a graph-based method is to yield a simpler but more lightweight solution.

Qualitative results The order matters for detecting key-value relationship, since the direction of a link induce a property for the destination entity that enrich its meaning. Differently from FUDGE [8] we do make use of directed edges, which led to a better understanding of the document having interpretable results. In Fig. 5 we show our qualitative results using Doc2Graph on groundtruth: green and red dots mean source and destination nodes, respectively. As shown in the different example cases, Fig. 5(a) and 5(b) resemble a simple structured form layout with directed one-to-one key-value association pairs and Doc2Graph manages to extract them. On the contrary, where the layout appears to be more complex as in Fig. 5(d), Doc2Graph fails to generalize the concept of one-to-many key-value relationship pairs. This may be due to the small number of trainable samples we had in our training data and the fact that header-cells usually present different positioning and semantic meaning. In the future we will integrate a table structure recognition path into our pipeline, hoping to improve the extraction of all kinds of key-value relationships in such more complex layout scenarios.

4.3 RVL-CDIP Invoices

Dataset In the work of Riba et al. [25] another subset of RVL-CDIP has been released. The authors selected 518 documents from the invoices classes, annotating 6 different regions (two examples of annotations are shown in Fig. 4). The task that can be performed are layout analysis, in terms of node classification, and table detection, in terms of bounding box (IoU > 50).

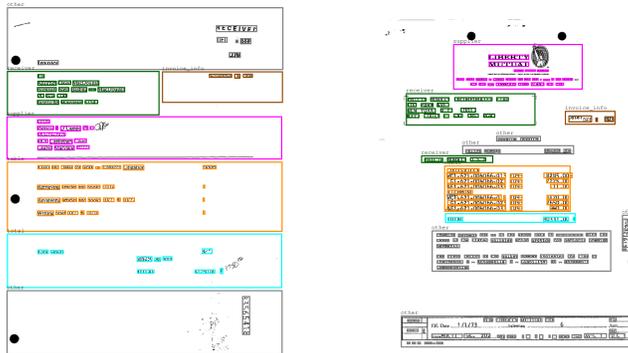


Fig. 4: **RVL-CDIP Invoices benchmark** in [25]. There are 6 regions: supplier (pink), invoice_info (brown), receiver (green), table (orange), total (light blue), other (gray).

Numerical results As done previously, we perform a k-fold cross validation keeping, for each fold, the same amount of test (104), val (52) and training documents (362). This time we applied an OCR to build the graph. There are two tasks: layout analysis, in terms of accuracy, and table detection, using F1 score and $IoU > 0.50$ for table regions. Our model outperforms [25] in both tasks, as shown in tables 4 and 5. In particular, for table detection, we extracted the subgraph induced by the edge classified as ‘table’ (two nodes are linked if they are in the same table) to extract the target region. Riba et al. [25] formulated the problem as a binary classification: we report, for brevity, in Tab. 5 the threshold on confidence score they use to cut out edges, that in our multi-class setting (‘none’ or ‘table’) is implicitly set to 0.50 by the softmax.

Qualitative results In Fig. 6 we show the qualitative results. The two documents are duplicated to better visualize the two tasks. For layout analysis, the greater boxes colors indicate the true label that the word inside should have (the colors reflects classes as shown in Fig. 4). For the table detection we use a simple

Method	Accuracy (\uparrow)	
	Max	Mean
Riba et al. [25]	62.30	-
Doc2Graph + OCR	69.80	67.80 \pm 1.1

Table 4: **Layout analysis results on RVL-CDIP Invoices.** Layout analysis accuracy scores depicted in terms of node classification task.

Method	Threshold	Metrics (\uparrow)		
		Precision	Recall	F ₁
Riba et al. [25]	0.1	0.2520	0.3960	0.3080
Riba et al. [25]	0.5	0.1520	0.3650	0.2150
Doc2Graph + OCR	0.5	0.3786 \pm 0.07	0.3723 \pm 0.07	0.3754 \pm 0.07

Table 5: **Table Detection in terms of F1 score.** A table is considered correctly detected if its IoU is greater than 0.50. Threshold values refers to the scores an edges has to have in order to do not be cut: in our case is set to 0.50 by the softmax in use.

heuristic: we take the enclosing rectangle (green) of the nodes connected by ‘table’ edges, then we evaluate the IoU with target regions (orange). This heuristic is effective but simple and so error-prone: if a false positive is found outside table regions this could lead to a poor detection result, e.g. a bounding box including also ‘sender item’ entity or ‘receiver item’ entity. In addition, as inferred from Figs. 6(a) and 6(b), ‘total’ regions could be taken out. In the future, we will refine this behaviour by both boosting the node classification task and including ‘total’ as a table region for the training of edges.

5 Conclusion

In this work, we have presented a task-agnostic document understanding framework based on a Graph Neural Network. We propose a general representation of documents as graphs, exploiting fully connectivity between document objects and letting the network automatically learn meaningful pairwise relationships. Node and edge aggregation functions are defined by taking into account the relative positioning of document objects. We evaluated our model on two challenging benchmarks for three different tasks: entity linking on forms, layout analysis on invoices and table detection. Our preliminary results show that our model can achieve promising results, keeping the network dimensionality considerably low. For future works, we will extend our framework to other documents and tasks, to deeper investigate the generalization property of the GNN. We would like to explore more extensively the contribution of different source features and how to combine them in more meaningful and learnable ways.

Acknowledgment

This work has been partially supported by the the Spanish projects MIRANDA RTI2018-095645-B-C21 and GRAIL PID2021-126808OB-I00, the CERCA Program / Generalitat de Catalunya, the FCT-19-15244, and PhD Scholarship from AGAUR (2021FIB-10010).

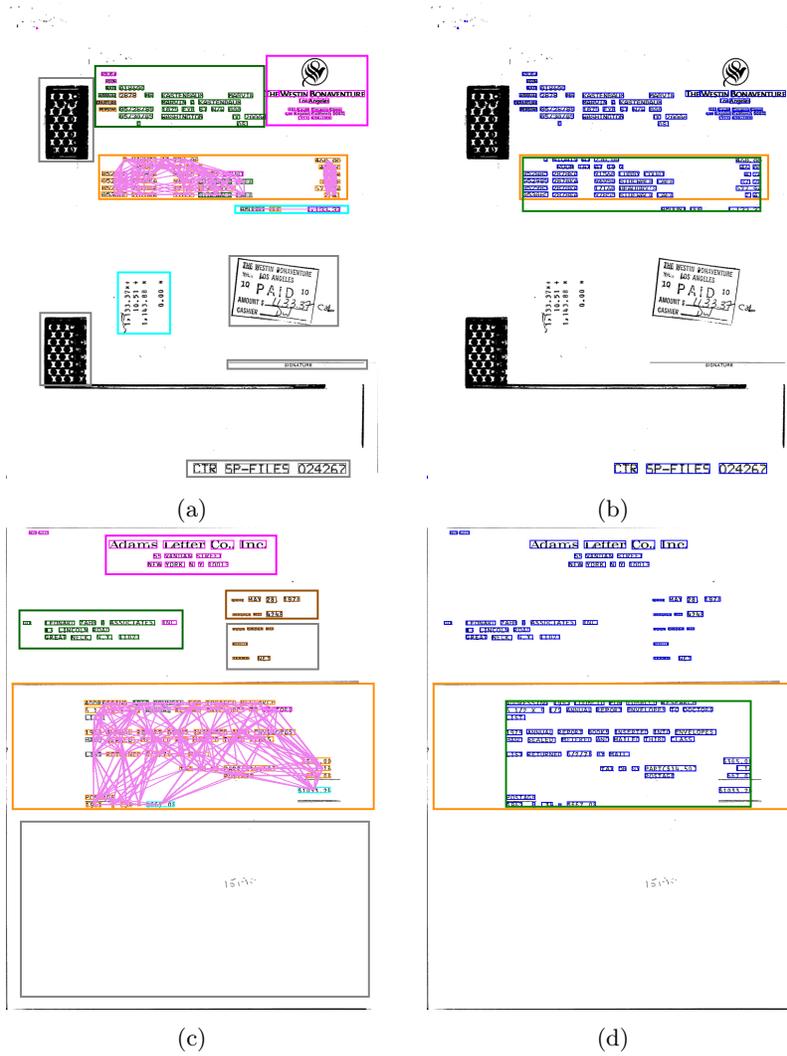


Fig. 6: **Layout Analysis on RVLCDIP Invoices.** Inference over two documents from RVL-CDIP Invoices, showing both Layout Analysis (a,c) and Table Detection (b,d) tasks.

References

1. Appalaraju, S., Jasani, B., Kota, B.U., Xie, Y., Manmatha, R.: Docformer: End-to-end transformer for document understanding. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 993–1003 (2021) [2](#), [3](#)
2. Biswas, S., Banerjee, A., Lladós, J., Pal, U.: Docsegtr: An instance-level end-to-end document image segmentation transformer. arXiv preprint arXiv:2201.11438 (2022) [3](#)
3. Biswas, S., Riba, P., Lladós, J., Pal, U.: Beyond document object detection: instance-level segmentation of complex layouts. International Journal on Document Analysis and Recognition (IJDAR) **24**(3), 269–281 (2021) [3](#)
4. Biswas, S., Riba, P., Lladós, J., Pal, U.: Graph-based deep generative modelling for document layout generation. In: International Conference on Document Analysis and Recognition. pp. 525–537. Springer (2021) [2](#)
5. Borchmann, L., Pietruszka, M., Stanislawek, T., Jurkiewicz, D., Turski, M., Szyn- dler, K., Graliński, F.: Due: End-to-end document understanding benchmark. In: Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2) (2021) [3](#)
6. Carbonell, M., Riba, P., Villegas, M., Fornés, A., Lladós, J.: Named entity recog- nition and relation extraction with graph neural networks in semi structured docu- ments. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 9622–9627. IEEE (2021) [2](#), [3](#), [9](#), [10](#)
7. Davis, B., Morse, B., Cohen, S., Price, B., Tensmeyer, C.: Deep visual template- free form parsing. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 134–141. IEEE (2019) [3](#)
8. Davis, B., Morse, B., Price, B., Tensmeyer, C., Wiginton, C.: Visual fudge: Form understanding via dynamic graph editing. In: International Conference on Docu- ment Analysis and Recognition. pp. 416–431. Springer (2021) [2](#), [3](#), [4](#), [7](#), [9](#), [10](#)
9. Gemelli, A., Vivoli, E., Marinai, S.: Graph neural networks and representation embedding for table extraction in PDF documents. In: accepted for publication at ICPR22 (2022) [2](#), [3](#)
10. Goldmann, L.: Layout Analysis Groundtruth for the RVL-CDIP Dataset (Sep 2019). <https://doi.org/10.5281/zenodo.3257319> [2](#), [3](#)
11. Hamilton, W., Ying, Z., Leskovec, J.: Inductive representation learning on large graphs. Advances in neural information processing systems **30** (2017) [6](#)
12. Harley, A.W., Ufkes, A., Derpanis, K.G.: Evaluation of deep convolutional nets for document image classification and retrieval. In: International Conference on Document Analysis and Recognition (ICDAR) [8](#)
13. Hong, T., Kim, D., Ji, M., Hwang, W., Nam, D., Park, S.: Bros: a pre-trained language model for understanding texts in document (2020) [3](#), [9](#), [10](#)
14. Huang, Z., Chen, K., He, J., Bai, X., Karatzas, D., Lu, S., Jawahar, C.: Icdar2019 competition on scanned receipt ocr and information extraction. In: 2019 Interna- tional Conference on Document Analysis and Recognition (ICDAR). pp. 1516– 1520. IEEE (2019) [3](#)
15. Jaume, G., Ekenel, H.K., Thiran, J.P.: Funsd: A dataset for form understanding in noisy scanned documents. In: 2019 International Conference on Document Analysis and Recognition Workshops (ICDARW). vol. 2, pp. 1–6. IEEE (2019) [2](#), [3](#), [8](#), [9](#)
16. Jocher, G., Chaurasia, A., Stoken, A., Borovec, J., NanoCode012, Kwon, Y., TaoXie, Fang, J., imyhxy, Michael, K.: ultralytics/yolov5: v6. 1-tensorrt, tensorflow edge tpu and opencv export and inference. Zenodo, Feb **22** (2022) [8](#), [9](#)

17. Li, X., Wu, B., Song, J., Gao, L., Zeng, P., Gan, C.: Text-instance graph: Exploring the relational semantics for text-based visual question answering. *Pattern Recognition* **124**, 108455 (2022) [2](#)
18. Liang, Y., Wang, X., Duan, X., Zhu, W.: Multi-modal contextual graph neural network for text visual question answering. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 3491–3498. IEEE (2021) [2](#)
19. Lin, T.Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., Dollár, P., Zitnick, C.L.: Microsoft coco: Common objects in context. In: European conference on computer vision. pp. 740–755. Springer (2014) [8](#)
20. Liu, H., Li, X., Liu, B., Jiang, D., Liu, Y., Ren, B.: Neural collaborative graph machines for table structure recognition. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4533–4542 (2022) [2](#), [3](#)
21. Mathew, M., Karatzas, D., Jawahar, C.: Docvqa: A dataset for vqa on document images. In: Proceedings of the IEEE/CVF winter conference on applications of computer vision. pp. 2200–2209 (2021) [3](#)
22. Powalski, R., Borchmann, L., Jurkiewicz, D., Dwojak, T., Pietruszka, M., Palka, G.: Going full-tilt boogie on document understanding with text-image-layout transformer. In: International Conference on Document Analysis and Recognition. pp. 732–747. Springer (2021) [3](#)
23. Qasim, S.R., Mahmood, H., Shafait, F.: Rethinking table recognition using graph neural networks. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 142–147. IEEE (2019) [3](#)
24. Raja, S., Mondal, A., Jawahar, C.: Table structure recognition using top-down and bottom-up cues. In: European Conference on Computer Vision. pp. 70–86. Springer (2020) [3](#)
25. Riba, P., Dutta, A., Goldmann, L., Fornés, A., Ramos, O., Lladós, J.: Table detection in invoice documents by graph neural networks. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 122–127. IEEE (2019) [2](#), [3](#), [4](#), [10](#), [11](#), [12](#)
26. Riba, P., Goldmann, L., Terrades, O.R., Rusticus, D., Fornés, A., Lladós, J.: Table detection in business document images by message passing networks. *Pattern Recognition* **127**, 108641 (2022) [2](#), [3](#)
27. Ronneberger, O., Fischer, P., Brox, T.: U-net: Convolutional networks for biomedical image segmentation. *CoRR* [abs/1505.04597](https://arxiv.org/abs/1505.04597) (2015), <http://arxiv.org/abs/1505.04597> [5](#)
28. Sarkar, M., Aggarwal, M., Jain, A., Gupta, H., Krishnamurthy, B.: Document structure extraction using prior based high resolution hierarchical semantic segmentation. In: European Conference on Computer Vision. pp. 649–666. Springer (2020) [3](#)
29. Singh, A., Natarajan, V., Shah, M., Jiang, Y., Chen, X., Batra, D., Parikh, D., Rohrbach, M.: Towards vqa models that can read. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition. pp. 8317–8326 (2019) [3](#)
30. Smock, B., Pesala, R., Abraham, R.: Pubtables-1m: Towards comprehensive table extraction from unstructured documents. In: Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition. pp. 4634–4642 (2022) [3](#)
31. Vu, H.M., Nguyen, D.T.N.: Revising funsd dataset for key-value detection in document images. *arXiv preprint arXiv:2010.05322* (2020) [8](#)

32. Xu, Y., Xu, Y., Lv, T., Cui, L., Wei, F., Wang, G., Lu, Y., Florencio, D., Zhang, C., Che, W., et al.: Layoutlmv2: Multi-modal pre-training for visually-rich document understanding. arXiv preprint arXiv:2012.14740 (2020) [2](#), [3](#)
33. Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M.: Layoutlm: Pre-training of text and layout for document image understanding. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining. pp. 1192–1200 (2020) [2](#), [3](#), [9](#), [10](#)
34. Xue, W., Yu, B., Wang, W., Tao, D., Li, Q.: Tgrnet: A table graph reconstruction network for table structure recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1295–1304 (2021) [2](#)
35. Yu, W., Lu, N., Qi, X., Gong, P., Xiao, R.: Pick: processing key information extraction from documents using improved graph learning-convolutional networks. In: 2020 25th International Conference on Pattern Recognition (ICPR). pp. 4363–4370. IEEE (2021) [2](#)