

50.035 Computer Vision 1D Project

Final Report

Team 8
Research Track

<https://github.com/tangh146/DETR-GFTE>

- 1. INTRODUCTION
- 2. METHODOLOGY
 - 2.1 DATASET SELECTION
 - 2.2 STAGED IMPROVEMENTS
 - 2.3 STAGE 1: ORIGINAL IMPLEMENTATION
 - 2.4 STAGE 2: ADAPTIVE WINDOW GRAPH
 - 2.5 STAGE 3: REPLACING RCNN WITH VIT
- 3. RESULTS
 - 3.1 INFERENCE EXAMPLES
 - 3.2 EVALUATION
- 4. DISCUSSION
 - 4.1 ATTENTION VISUALIZATION
- 5. CONCLUSION
- 6. APPENDIX
 - 6.1 APPENDIX A
 - 6.2 APPENDIX B
- 7. REFERENCES

1. INTRODUCTION

Table recognition models convert the input image of a table into a machine-understandable format, such as HTML or adjacency matrices. These models are critical for applications like RAG and LLM dataset curation.

While most table recognition models ^[1,2] rely on a Seq2Seq architecture to convert tables into HTML format, they have some shortcomings as compared to Graph Convolutional Networks (GCNs) ^[3]. The latter approach offers several advantages, including the mitigation of cumulative autoregression errors (see Appendix A) and the potential to address non-Euclidean challenges, such as recognizing flowcharts and other complex structures. However, the GCN approach remains relatively underexplored, with a notable scarcity of implementations.

Graph-based Financial Table Extraction (GFTE) is a well-cited research paper and an archetypal implementation of the graph-based table recognition model. The focus of this project will be to enhance the current GFTE model by introducing architectural modifications to improve its performance.

2. METHODOLOGY

2.1 DATASET SELECTION

The original GFTE model is trained and evaluated on the SciTSR and FinTab datasets ^[3], which have become obsolesced by newer datasets such as PubTabNet ^[4].

Dataset	Origin	Size
SciTSR	LaTeX source files of scientific papers (English)	15,000
FinTab	Financial reports (Chinese) (PDFs)	1,685
PubTabNet	XML papers in the PubMed Central Open Access (English)	568,000

As a much larger dataset, PubTabNet is more diverse in the style and content of tables, offering better generalizability to real-life documents. As of 2024, it is the most comprehensive, state-of-the-art table dataset on the market.

2.2 STAGED IMPROVEMENTS

Our methodology is to first implement the original GFTE model as per the authors' specifications, modernizing it on the PubTabNet dataset. Then, we identify problems and incrementally replace individual components to improve the model.

2.3 STAGE 1: ORIGINAL IMPLEMENTATION

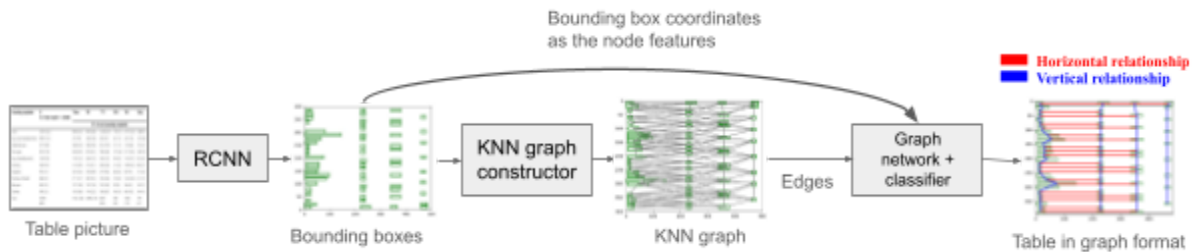


Fig 1. Outline of GFTE's original implementation.

While the original paper details variants of GFTE that involve feature extraction using NLP, we

implement the fundamental version which captures the core ideas behind GCN-based table extraction. GFTE works by first extracting the cell bounding boxes from the image, before using their coordinates to construct a k-Nearest Neighbor (KNN) graph that aims to map out all candidate edges in the table. Candidate edges connect groups of cell bounding boxes together and may or may not delineate structural relationships between them. The edges and bounding box coordinates are delivered to a GCN-driven classification head that classifies the edges into one of three classes— either horizontal relationship, vertical relationship or no relationship— thereby constructing the finished table.

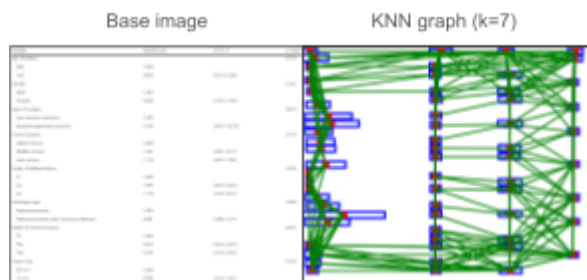


Fig 2. KNN graph construction.

Fig 2 shows the outcome of constructing a KNN graph from a PubTabNet sample. The new dataset exposes the inadequacies of using a basic KNN algorithm to draw the candidate edges. This sample contains a lot of tightly clustered cells and dead spaces between the columns. In particular, the cells are much closer to their columnar neighbors than to the cells on their left and right. This causes the algorithm to oversample neighbors from the same column while neglecting those from the same row, causing dead spaces in the resulting graph and potentially leading to imbalance in training.

2.4 STAGE 2: ADAPTIVE WINDOW GRAPH

Variable	Male		Female	
	Adaptive window	% CI	%	95% CI
Sensitivity	95.17	93.95 to 96.32	97.94	96.49 to 99.32
Specificity	93.92	92.42 to 95.42	92.42	91.18 to 93.64
Positive Likelihood Ratio	6.43	4.95 to 8.43	4.95	3.89 to 6.29
Negative Likelihood Ratio	0.65	0.57 to 0.73	0.68	0.60 to 0.76
Disease prevalence	8.61	7.37 to 9.97	8.36	7.23 to 9.59
Positive Predictive Value	97.94	96.95 to 98.94	94.88	93.04 to 96.63
Negative Predictive Value	94.25	93.04 to 95.31	94.19	93.07 to 95.18

Fig 3. Adaptive window on a “horizontally-stretched” table.

To alleviate the problem identified above, we propose an “adaptive window” graph constructor which takes into account the dead space between the cells to adjust the window within which

neighbors are sampled. This reduces oversampling from a single axis.

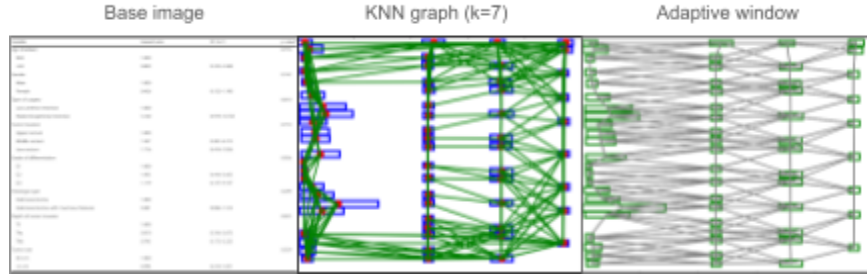


Fig 4. Comparison of graph constructors.

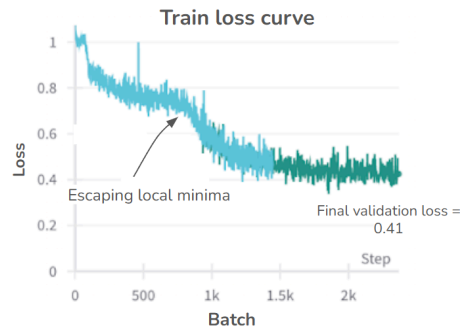


Fig 5. Train loss curve.

Regardless of hyperparameter tuning, local minima will manifest at some point along the train loss curve (Fig 5). This shows that the loss landscape is very rough. We surmise that this is because the feature vectors (made up of normalized bounding box coordinates) are not expressive enough to capture the cells and their spatial relationships.

2.5 STAGE 3: REPLACING RCNN WITH ViT

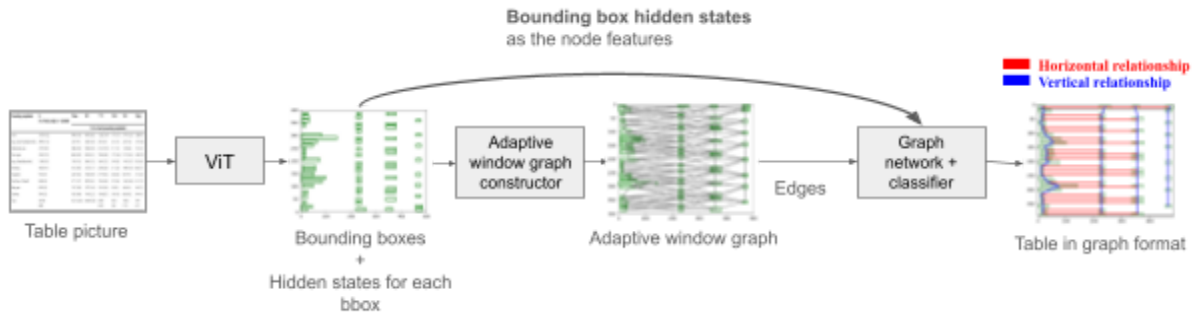


Fig 6. Outline of the improvements on GFTE.

To improve the expressiveness of the feature vectors, we replaced the R-CNN object detector

with the ViT-based object detector DETR ^[5]. DETR produces a hidden state for every bounding box, allowing us to leverage the hidden states as node features for the graph network. The ViT is trained with the object label for any particular box being the number of edges connected to it. Our hypothesis is that this will enrich the hidden states ($dim = 256$) with more spatial information of the box's immediate surroundings and distant relationships such as with column headers. The hypothesis assumes that the attention mechanism in the ViT will learn meaningful interpatch and interbox relationships.

To accommodate the large increase in the feature vector dimension (from $dim = 8$ to $dim = 256$), the hidden dimension of the graph-based classification head is increased and an additional GCN layer is supplemented (Fig 7).

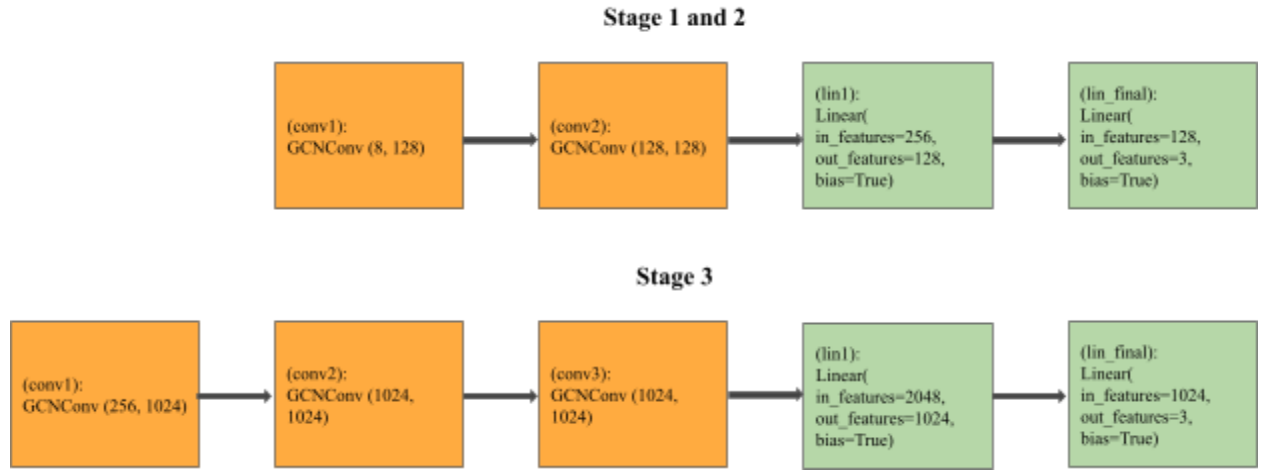


Fig 7. Increased depth in Stage 3.

3. RESULTS

3.1 INFERENCE EXAMPLES

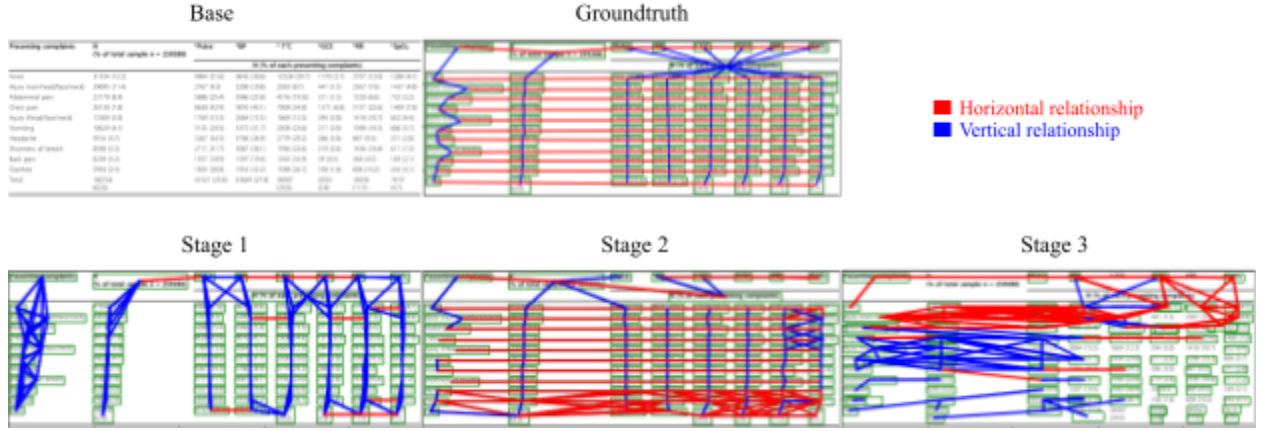


Fig 8: Forward passes with a sample image on all three models.

To reiterate and analyze the shortcomings of each model, we pass a sample table through each model. Stage 1 demonstrates a clear overrepresentation of columnar relationships due to the limitations of the KNN graph constructor. Even the comparatively dense right-hand side of the table exhibits few horizontal relations. This suggests that oversampling of vertical relations is causing a training imbalance, causing the model to disproportionately classify edges as vertical relations. Stage 2 shows a much more balanced table due to the improved graph constructor. However, it is clear that merged cells are still a problem. The result of Stage 3 is incomprehensible, with no discernible pattern.

3.2 EVALUATION

The authors of GFTE calculate evaluation accuracy based on the number of correct predictions. However, this does not take into account the accuracy of the graph constructor, which may miss edges or generate redundant edges, causing cardinality mismatch between the constructed edges and the groundtruth edges. A simple calculation $Accuracy = \frac{n(\text{correctly classified edges})}{n(\text{predicted edges})}$ will thus be insufficient for our task. Instead, we propose a modified calculation to ensure that the numerator only counts the constructed edges that are correctly classified **AND** exist within the groundtruth edge set:

$$Accuracy = \frac{n(\text{correctly classified edges} \cap \text{groundtruth edges})}{n(\text{groundtruth edges})}$$

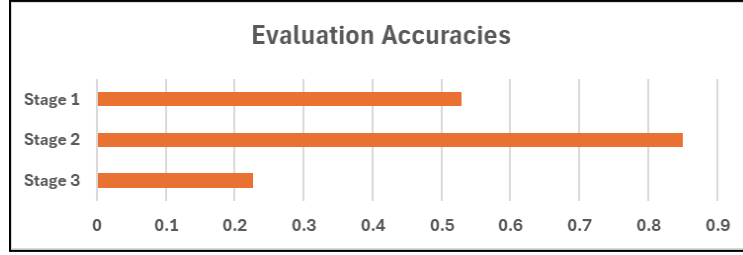


Fig 9: Accuracies derived from our evaluation set ($n = 50$) using our custom evaluation metric.

4. DISCUSSION

4.1 ATTENTION VISUALIZATION

The subpar performance of our Stage 3 model prompts us to investigate why the DETR-produced hidden states are unable to properly capture the spatial information of the underlying table cells. Our initial hypothesis is that the attention mechanism of DETR is able to learn meaningful interpatch and interbox relationships. In the DETR model, the encoder processes interpatch attention while the decoder processes interbox attention ^[5]. We will be analyzing the decoder attentions.

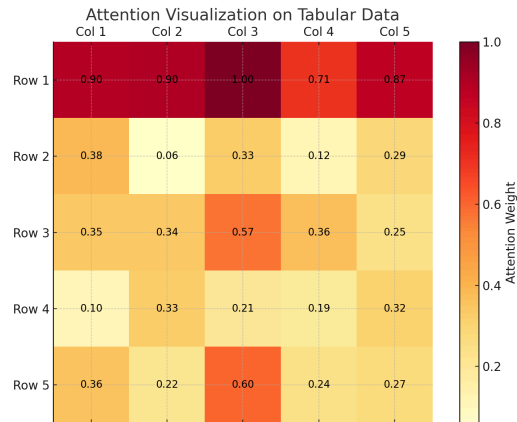


Fig 10: Decoder (interbox) attentions with respect to Row 1, Col 3.

The attention visualisations, represented as heatmaps, reveal the areas where the model concentrates its focus while processing the table. These visualisations highlight the intensity of attention weights assigned to different cells or bounding boxes, variations in attention across

rows and columns, providing insights into how the model interprets table relationships, patterns of attention in specific scenarios, such as densely clustered bounding boxes or tables with significant empty spaces. For example, the heatmaps demonstrate that the model assigns higher attention weights to cells containing headers and key data points, ensuring that these critical elements influence the final predictions more significantly. (Kasem et al., 2022)

The model effectively prioritizes bounding boxes corresponding to headers and key data entries, which serve as anchors for logical relationships within the table. In tables with unbalanced structures or merged cells, attention tends to diffuse, leading to weaker connections between nodes. The visualizations show stronger attention in horizontal relationships, reflecting the model's reliance on row-wise dependencies. However, vertical relationships may be underemphasized in certain cases.

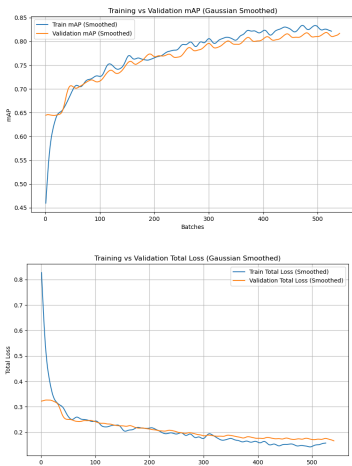
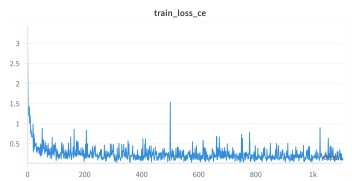
The attention mechanism has both advantages and weaknesses that impact the results. The advantage is by focusing on critical regions of the table, attention improves the alignment of predicted relationships with the ground truth. This contributes to higher accuracy, particularly in scenarios with well-structured tables. The weakness is in cases where the visual and logical relationships are highly irregular, the attention mechanism struggles to distribute focus effectively, resulting in misaligned predictions and lower accuracy.

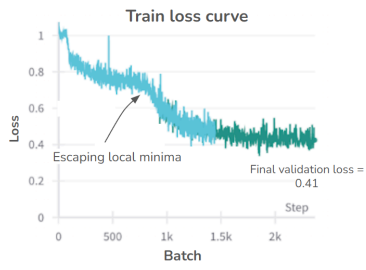
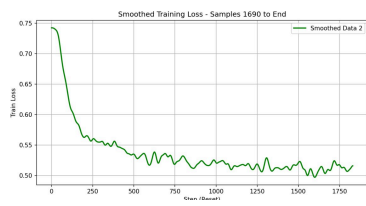
5. CONCLUSION

This work demonstrates the progression of improvements on the GFTE model. While the original GFTE authors have successfully used NLP techniques to extract textual features from the cells to improve feature expressiveness, we investigate the effectiveness of using DETR ViT for that purpose. More work needs to be done to optimize feature extraction in graph-based table recognition models.

In future work, we aim to develop more sophisticated methods for feature extraction, enabling richer representations of node features. Specifically, integrating advanced pre-trained models or custom encoders for domain-specific tasks could further enhance the performance. Additionally, the adaptability of the graph-based framework will be explored by extending it to handle more complex structured data, such as flowcharts, which present unique non-Euclidean

6.2 APPENDIX B

Model	Training curves	Metrics	Hyperparameters
FasterRCNN	 <p>Every validation batch average results from Epoch 1 to 15 is plotted. But only every 55th training batch average results from Epoch 1 to 15 is plotted.</p>	<p>Validation Last Epoch Batch Average Accuracies and losses:</p> <p>avg_loss_classifier : 0.0426</p> <p>avg_loss_box_reg : 0.1002</p> <p>avg_loss_objectness : 0.0034</p> <p>avg_loss_rpn_box_reg : 0.0267</p> <p>avg_total_loss : 0.1729</p> <p>avg_precision : 0.9898</p> <p>avg_recall : 0.9379</p> <p>avg_f1_score : 0.9550</p> <p>avg_mAP : 0.8137</p>	<p>batch size=32</p> <p>num epochs = 15</p> <p>base learning rate = 1e-4</p> <p>lr = base_lr * 8 (scaled learning rate for 8 GPUs)</p> <p>Adam W optimizer: weight_decay=0.01</p> <p>lr = lr</p> <p>Scheduler: OneCycleLR with parameters:</p> <p>max_lr = lr</p> <p>total_steps = num_batches * num_epochs=29355</p> <p>pct_start=0.1</p> <p>Gradient Accumulation steps = 4</p>
detr-resnet-50		<p>avg_loss_classifier : 0.085</p> <p>avg_loss_bbox : 0.195</p> <p>avg_loss_giou : 0.107</p> <p>avg_total_loss : 0.387</p> <p>avg_precision : 0.692</p> <p>avg_recall : 0.656</p> <p>avg_f1_score : 0.673</p> <p>avg_mAP : 0.569</p>	<p>devices: 1</p> <p>val_check_interval: 0.5</p> <p>gradient_clip_val: 0.1</p> <p>accumulate_grad_batches: 12</p>
GCN Stage 1		<p>Validation_loss: 0.34</p> <p>Evaluation_accuracy: 0.529</p>	<p>d_model: 128</p> <p>lr: 1e-3</p> <p>batch_size: 32</p> <p>num_workers: 8</p> <p>devices: 1</p> <p>val_check_interval: 0.3</p>

GCN Stage 2		Validation_loss: 0.41 Evaluation_accuracy: 0.850	d_model: 128 lr: 1e-3 batch_size: 32 num_workers: 8 devices: 1 val_check_interval: 0.3
GCN Stage 3		validation_loss: 0.509 evaluation_accuracy: 0.226	d_model: 1024 lr: 1e-3 batch_size: 32 num_workers: 8 devices: 1 val_check_interval: 0.3

7. REFERENCES

- [1] ShengYun Peng, Aishwarya Chakravarthy, Seongmin Lee, Xiaojing Wang, Rajarajeswari Balasubramanian, & Duen Horng Chau. (2024). UniTable: Towards a Unified Framework for Table Recognition via Self-Supervised Pretraining.
- [2] Chenxia Li, Ruoyu Guo, Jun Zhou, Mengtao An, Yuning Du, Lingfeng Zhu, Yi Liu, Xiaoguang Hu, & Dianhai Yu. (2022). PP-StructureV2: A Stronger Document Analysis System.
- [3] Yiren Li, Zheng Huang, Junchi Yan, Yi Zhou, Fan Ye, & Xianhui Liu. (2020). GFTE: Graph-based Financial Table Extraction.
- [4] Zhong, X., Shafiei Bavani, E., & Jimeno Yepes, A. (2020). Image-based table recognition: Data, model, and evaluation. *Lecture Notes in Computer Science*, 564–580.
- [5] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, & Sergey Zagoruyko. (2020). End-to-End Object Detection with Transformers.
- [6] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention is All you Need. *arXiv (Cornell University)*, 30, 5998–6008. <https://arxiv.org/pdf/1706.03762v5>
- [7] Kasem, M., Abdallah, A., Berendeyev, A., Elkady, E., Abdalla, M., Mahmoud, M., Hamada, M., Nurseitov, D., & Taj-Eddin, I. (2022). Deep learning for table detection and structure recognition: A survey. *arXiv preprint arXiv:2211.08469*. <https://arxiv.org/pdf/2211.08469>