INDRAPRASTHA INSTITUTE *of*
INFORMATION TECHNOLOGY **DELHI**

Detection to Interpretation: Advancing Tabular Data Processing with Multimodal AI

A Thesis Report

submitted by

PIJUSH BHUYAN
MT22049

in partial fulfillment of the requirements
for the award of the degree of

MASTER OF TECHNOLOGY
in
COMPUTER SCIENCE AND ENGINEERING
(Specialization in Artificial Intelligence)

INDRAPRASTHA INSTITUTE OF INFORMATION TECHNOLOGY DELHI
NEW DELHI - 110020

21st December 2024

THESIS CERTIFICATE

This is to certify that the thesis titled ***Detection to Interpretation: Advancing Tabular Data Processing with Multimodal AI,*** submitted by **Pijush Bhuyan**, to the Indraprastha Institute of Information Technology, Delhi for the award of the degree of Master of Technology is a bonafide record of the research work done by him under my supervision. This thesis has reached the standards of fulfilling the requirements of the regulations relating to the degree.

The contents of this thesis, whether in full or in parts, have not been submitted to any other Institute or University for the award of any degree or diploma.

Rajiv Ratn Shah

Dr. Rajiv Ratn Shah
Thesis Supervisor
Associate Professor and Institute Chair Professor
Department of CSE and HCD
Indraprastha Institute of Information Technology, Delhi

Place : New Delhi, India
Date : 21st December 2024

ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to all those who have supported and guided me throughout my M.Tech thesis work. First and foremost, I would like to thank **Dr. Rajiv Ratn Shah** for unwavering support and belief in me, which has provided me great strength, encouragement and patience to carry out this research work. I am also very grateful to **Professor Shin'ichi Satoh**, for giving me the opportunity to collaborate with and continue this work at the National Institute of Informatics Japan, during the period of my research internship.

A special thanks to ***Avinash Anand***, who introduced me to this domain, which motivated me to continue working and take up this problem statement for my thesis. I am very glad to collaborate with the team - ***Raj Jaiswal, Mohit Gupta, Janak Kapuriya, and Debnath Kundu***. Our initial work together has greatly helped me to further develop this problem statement.

I am also very thankful to the members of ***MIDAS Labs (IIIT Delhi)*** and ***Satoh-Lab (NII Japan)***, whose valuable feedback and suggestions during the team meetings helped me to brainstorm novel ideas to apply for this project.

Lastly, I would like to thank the institutes - *Indraprastha Institute of Information Technology, Delhi* and *National Institute of Informatics, Japan* for providing me with the opportunity, infrastructure and computational resources required for carrying out this research work.

PIJUSH BHUYAN
MT22049
COMPUTER SCIENCE AND ENGINEERING

Abstract

Tables are the most common form of structured data found in documents. Proper interpretation of such raw tabular data by computer systems remains an open challenge. We take a deep dive into document intelligence - which includes table detection, table reconstruction and table structure interpretation by AI models.

Firstly, we handle domain adaptation in table detection. Pre-trained table detection models have displayed poor results when the target domain varied from the source. We resolve this by building a domain invariant table detection dataset where we inject additional noisy synthetic detection data. Empirical tests show that training a detection model on synthetic data displays a significantly lower drop in performance when tested on out-of-distribution datasets.

Following this, we build a fast,yet efficient, end-to-end pipeline for Table-OCR. It reconstructs the table structure and content from raw detection crops and converts them into computer-storable text format.

Finally, we design a comprehensive benchmark suite of tests to test the table structure understanding capabilities and limitations of existing Large Language models (LLMs) and Vision Language Models (VLMs) using both text and image modalities. The vision component of VLMs is found to be a bottleneck in multi-modal table interpretability. We work with a light-weight, yet efficient model-agnostic adapter module which injects positional information into the image modality through positional embeddings during model training. We also design a novel pre-training task for image-text alignment for open-source VLMs and study the change in model performance while interpreting visual tabular data. We also study the feasibility and future scope for true multimodal table understanding - interpreting tabular data from both image and text modalities for reasoning.

Keywords : *Table Detection, Unsupervised Domain Adaptation, Table Structure and Content Recognition (Table OCR), Table Question Answering, Table Structure Understanding, Multimodal Table Understanding, Large Language Models, Vision Language Models, Positional Insert Embeddings, Pretraining.*

Table of Contents

List of Figures

List of Tables

Chapter 1

Table detection and its challenges with varying domains

1.1 Table Detection and Domain Shift

There is a big demand in the industry for large-scale data extraction from documents to inform decisions, with layout driving extraction performance. Tables are the most popular format of representing structured data and tabular data in such documents which is ubiquitous all over the internet. It offers valuable content representation, enhancing the predictive capabilities of various systems such as search engines and Knowledge Graphs.

In Document Layout analysis, the automatic recognition of tabular data in document images presents a significant challenge due to the diverse range of table styles and complex structures. Pretrained layout detector models demonstrate subpar performance when tested on documents from diverging domains which have quite contrasting layouts. This is called the domain shift problem. It is very labor intensive and time consuming to manually annotate the data from a new domain and then finetune the existing detectors.

1.2 Domain Adaptation

Domain generalization and adaptation are distinct concepts in machine learning, primarily in supervised learning. Both aim to improve model performance on new data, but they vary in goals. Generalization involves excelling on unseen data from the same distribution as training data. Meanwhile, domain adaptation transfers knowledge from one domain to another with dissimilar distributions, aiming to enhance the model's target domain performance using labeled data from the source domain.

Traditional domain adaptation studies have introduced domain adaptation methods like feature selection, transfer learning, and domain-specific knowledge integration. However these approaches often face constraints in scalability, performance, or domain-specific applicability. The need for an effective solution to these limitations remains significant. The main goal is to enhance the model's ability to work well across different document layouts.

1.3 Synthetic Data and Unsupervised Domain Adaptation

Synthetic data has often been used to improve the generalization capabilities of machine learning models in the past. Whether it is about generating synthetic data with GANs and Diffusion models for training object detection models, or generating synthetic prompts from GPT to train open source instruction-tuned models like LLaVA [1], such unsupervised approaches have been proven beneficial for enhancing generalization capabilities of machine learning models.

We follow a similar approach to train generalized table detector models using noisy semi- synthetic training data.

1.4 Layout Detection Datasets

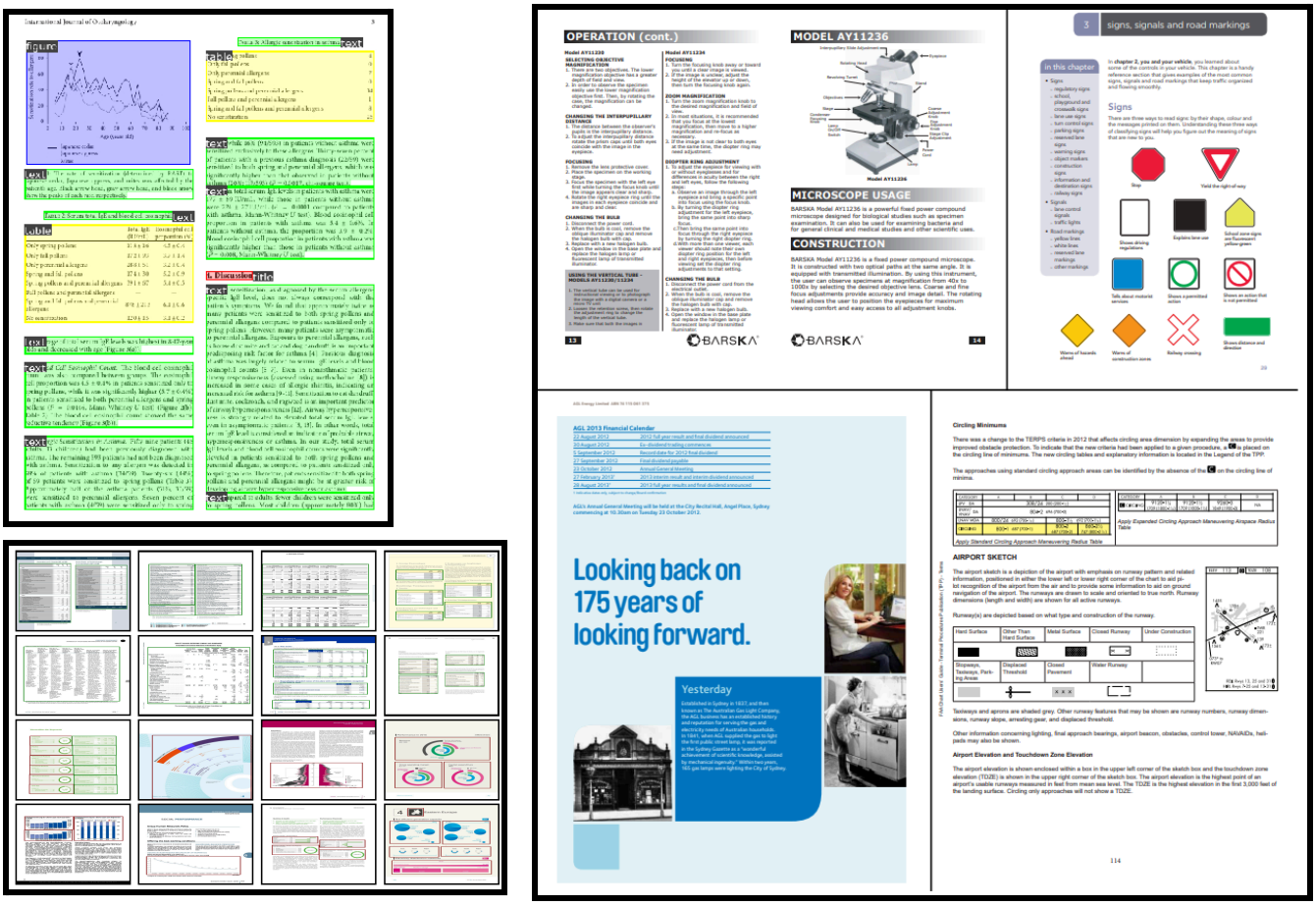


Figure 1 - Sample annotations from public table detection datasets. From top to bottom, left and right order : PubLayNet [2], Doclaynet [4] and IIITAR-13K [3]

1.4.1 PubLayNet - PubLayNet [2] is a massive scale document layout dataset containing over *360K* document images scraped from *PubMed Central Open Access* PDF documents annotated with bounding boxes and polynomial segmentations of text, title, lists, table and Figures classes. It contains layout structure for documents from the medical domain from the scientific community.

1.4.2 IIITAR-13K - IIITAR-13K [3] is another domain specific layout detection dataset created from the annual reports of diverse companies, which are a distinct category of business documents.They randomly choose publicly accessible annual reports in English and various other languages, including French, Japanese,Russian, among others, from over ten years of twenty-nine distinct companies. They perform manual annotation of bounding boxes for five distinct categories of graphical elements that are commonly present in annual reports: tables, Figureures, natural images, logos, and signatures. The IIIT-AR-13K dataset comprises *13,000* annotated pages containing *16,000 tables, 3,000 Figureures, 3,000 natural images, 500 logos, and 600 signatures*.

1.4.3 Doclaynet - Doclaynet [4] is a diverse layout detection dataset created from scraping document PDFs from 5 diverse domains - *financial documents, manuals, law and regulations, scientific* documents and patents. It contains over 86k manually annotated document images with bounding boxes for *11 classes* namely - *Caption, Footnote, Formula, List-item, Page-footer, Page-header, Picture, Section-header, Table, Text, and Title*.

Visualization of the annotations for all these three datasets are shown in Figure 1.

1.5 Layout Detector -

YOLO [5] was a breakthrough in real-time object detection when it was released, and it is still one of the most used models in practical vision applications. It moved the detection process from a two or three-stage process (i.e., R-CNN [6], Faster R-CNN [7]) to a single-staged convolutional stage and outperformed in both accuracy and speed compared to all the state-of-the-art object detection methods. The model architecture from the original paper has changed over time by adding different hand-crafted features to improve the model's accuracy. We take the *8th generation* for conducting our experiments, i.e. ***YOLO-v8.***

1.6 Methodology - Creating a Synthetic Dataset

Our objective is to perform domain adaptation in a self supervised manner and build a synthetic noisy dataset that can facilitate the development of highly resilient and versatile layout detection models which ideally should possess the ability to effectively handle various structures and formats. This innovative approach will enhance the model versatility while minimizing bias.

Our approach begins by utilizing the source dataset PubLayNet, which consists of fixed labels - *Text, Title, List, Figureure, and Table*. This dataset is employed for the initial training of our model. Subsequently, a target dataset is meticulously curated, characterized by an expanded class structure and diverse domains. The central challenge in domain adaptation lies in addressing the disparities between the source and target datasets.

Our **RanLayNet** breaks from conventional datasets by embracing diverse layout configurations, fostering adaptability, and reducing bias. It's a vital asset for domain adaptation. Starting with the source dataset and its fixed five labels, we initiate model training. This baseline performance serves to reveal limitations in adapting to the target dataset's diverse domains and extended class structure. Model evaluation comes next, as we test inference on the target dataset, revealing gaps in generalization. An essential aspect of this approach is the gradual expansion of the RanLayNet dataset. We methodically add diverse layout structures and distinct data distributions as patches onto the dataset canvas. This augmentation boosts dataset complexity and variance, exposing the model to a wider range of scenarios.

The process of creating a RanLayNet dataset with consolidated images and corresponding label information involved the following steps -

Step 1 - Build a csv file containing details about the images and their associated bounding boxes.
Step 2 - Start with a blank canvas and a composite document image.
Step 3 - Leveraging the *smart_plo*t function, positions and gaps are determined to accommodate each image or crop effectively within a composite image.Upon successful position and gap calculation, a blank canvas is generated as the base for the composite image.
Step 4 - Subsequently, the function iterates through each determined position. Based on the provided data, it selectively crops specific regions from the images or pastes entire images onto the composite canvas at the designated positions.

Step 5 - As this process unfolds, the coordinates of bounding boxes are updated, and these new coordinates are stored in a label file.

Iterating through all positions, the function orchestrates the arrangement and placement of images or crops onto the composite canvas, creating a comprehensive representation. Ultimately, the output comprises the composite image itself, and a label file containing adjusted bounding box information. This holistic approach streamlines the compilation of a RanLayNet dataset that offers a consolidated view of images alongside their corresponding labels, facilitating streamlined analysis and evaluation.

The visual representation of the whole pipeline is shown in Figure. 2

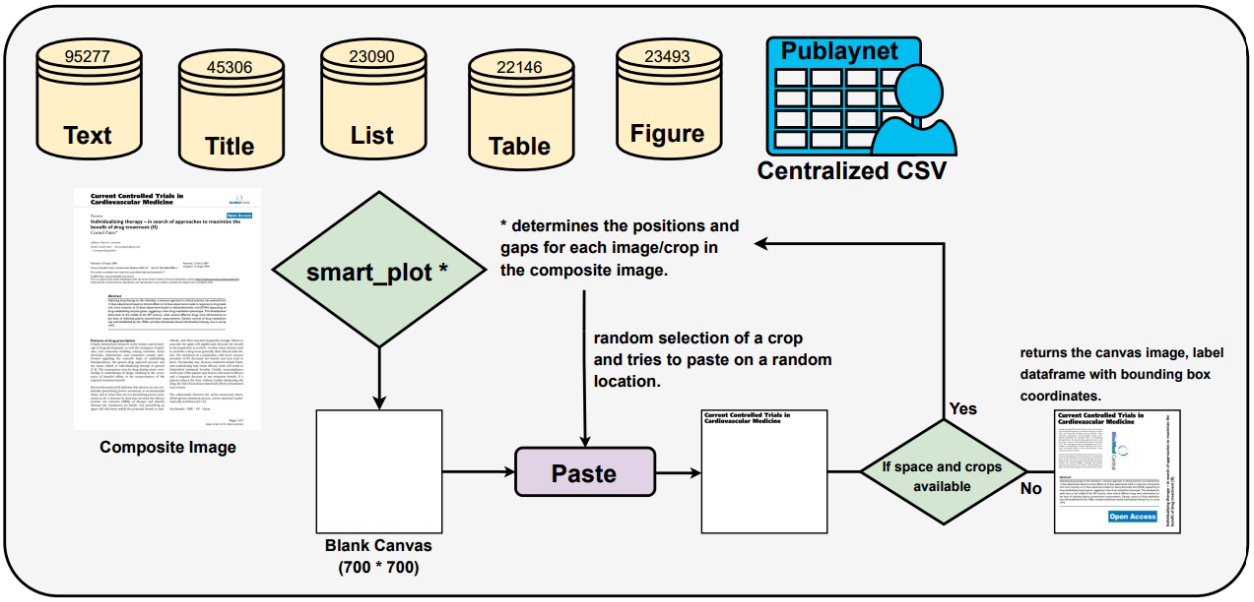


Figure 2 - the pipeline for creating synthetic RanLayNet from source Publayne images and labels.

The class distribution of the various layout components of RanLayNet is shown in Table 1. It should be noted that the Table class constitutes just about 11% of the entire annotation data. A sample from the synthetically generated dataset is shown in Figure 3. All the components are arranged randomly in an efficient manner on the blank canvas.

Label	Count	Percentage
Text	95,227	45.52
Title	45,306	21.65
List	23,090	11.03
Table	22,146	10.58
Figureure	23,493	11.22

Table 1 - Class distribution of the layout components of RanLayNet.

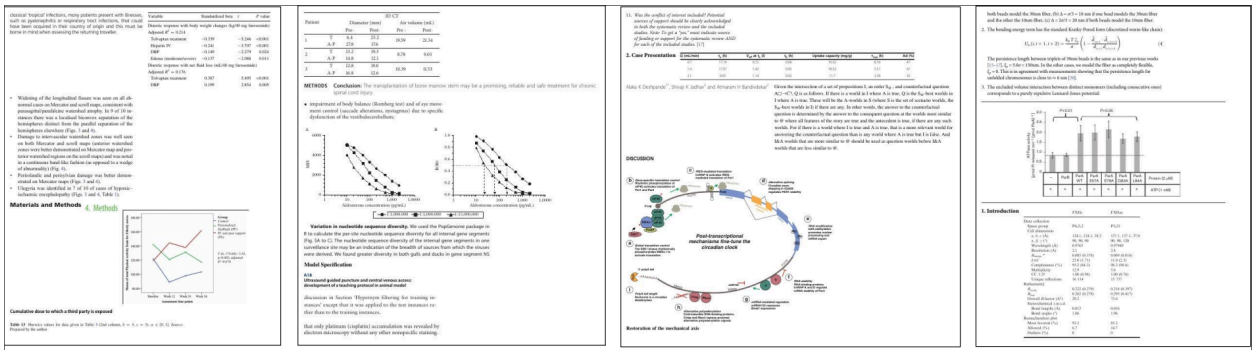


Figure 3 - A few samples of synthetic images created for RanLayNet

1.7 Experimentation

We perform two parallel sets of experiments for testing our hypothesis for domain adaptation. We take both PubLayNet and IIITAR-13K as our source domain class and Doclaynet as the target domain class. It is to be noted that the target dataset is quite diverse and contains documents from non-overlapping domains. Pretrained YoLo-v8 models are trained on the two clean source domains separately for 40 epochs with a *batch size of 16*, *SGD optimizer* and a *learning rate of 1e-3*. We test both these trained detectors on the Doclaynet dataset for the table class to test for their generalizability.

After this, we take both these models and further finetune them on our synthetic RanLayNet dataset containing noisy layout data and test both of them on the same Doclaynet images.

The evaluation metrics used for our case of table detection are ***Precision, Recall, mAP@50 and mAP@50:95.***

1.8 Results and Discussion

Domain	Pretrained on IIIT-AR-13K				Pretrained on IIIT-AR-13K + RanLayNet			
	Precision	Recall	mAP@50	mAP@0.5:95	Precision	Recall	mAP@50	mAP@0.5:95
Manuals	0.194	0.035	0.059	0.026	***0.446 (+0.252)***	***0.356 (+0.321)***	***0.336 (+0.277)***	***0.226 (+0.2)***
Financial	0.004	0.01	0.0003	0.00008	***0.374 (+0.37)***	***0.334 (+0.324)***	***0.298 (+0.297)***	***0.187 (+0.187)***
Laws and Regulations	0.194	0.353	0.0587	0.0262	***0.619 (+0.425)***	***0.263***	***0.293 (+0.234)***	***0.222 (+0.196)***
Scientific	0.01	0.004	0.0059	0.0022	***0.187 (+0.177)***	***0.474 (+0.44)***	***0.334 (+0.328)***	***0.298 (+0.296)***

Table 2 - DocLaynent inference results using Yolo-v8 pre-trained on IIIT-AR-13K, with and without our synthetic RanLayNet dataset

Domain	Pretrained on PubLayNet				Pretrained on PubLayNet + RanLayNet			
	Precision	Recall	mAP@50	mAP@0.5:95	Precision	Recall	mAP@50	mAP@0.5:95
Manuals	0.488	0.582	0.421	0.220	***0.562 (+0.074)***	***0.807 (+0.225)***	***0.761 (+0.34)***	***0.588 (+0.368)***
Financial	0.350	0.452	0.288	0.163	***0.427 (+0.077)***	***0.555 (+0.103)***	***0.465 (+0.177)***	***0.293 (+0.13)***
Laws and Regulations	0.039	0.356	0.337	0.194	***0.294 (+0.255)***	***0.572 (+0.216)***	***0.350 (+0.013)***	***0.282 (+0.088)***
Scientific	0.366	0.621	0.562	0.376	***0.562 (+0.196)***	***0.807 (+0.186)***	***0.761 (+0.199)***	***0.588 (+0.212)***

Table 3 - DocLaynent inference results using Yolo-v8 pre-trained on PubLayNet, with and without our synthetic RanLayNet dataset

Pre-training Dataset	Class	Precision	Recall	(m)AP@50	(m)AP@0.5:95
IIT-AR-13K	Logo	0.863	0.845	0.89	0.661
	Table	0.984	0.988	0.992	0.945
	Figureure	0.951	0.869	0.945	0.763
	Natural Image	0.95	0.936	0.968	0.919
	Signature	0.981	0.92	0.992	0.763
	All	0.946	0.922	0.957	0.81
PubLayNet	Figureure	0.933	0.884	0.931	0.894
	Text	0.933	0.785	0.924	0.870
	Title	0.843	0.958	0.886	0.705
	List	0.914	0.780	0.895	0.832
	Table	0.955	0.942	0.988	0.951
	All	0.910	0.880	0.914	0.855

Table 4 - Evaluation metrics on source datasets : IIIT-AR-13K and PubLayNet

Fine-tuning Dataset	Precision	Recall	(m)AP@50	(m)AP@0.5:95
All	0.968 / 0.925	0.945 / 0.884	0.974 / 0.934	0.918 / 0.858
Text	0.958 / 0.946	0.935 / 0.794	0.978 / 0.948	0.927 / 0.873
Title	0.949 / 0.843	0.981 / 0.966	0.98/ 0.903	0.80 / 0.711
List	0.956 / 0.922	0.908 / 0.795	0.971 / 0.896	0.944 / 0.842
Table	0.988 / 0.969	0.988 / 0.965	0.994 / 0.985	0.99 / 0.958
Figureure	0.989 / 0.947	0.912 / 0.90	0.947 / 0.940	0.931 / 0.904

Table 5 - Evaluation metrics, after fine-tuning pretrained models (PubLayNet/IIIT-AR-13K) with our RanLayNet dataset.

The initial pre-training results on the test sets of the source datasets and further fine-tuning on RanLayNet are reported in Tables 4 and 5 respectively. All the models show promising detection results on the source. In tables 2 and 3, we report the evaluation results after the domain shift using the models pretrained IIIT-AR-13K and PubLayNet respectively.

The initial inference results on the Doclaynet dataset using the layout detectors pretrained on clean data display poor generalizability as expected. Financial documents and scientific domains of Doclaynet display the worst reduction in performance with nearly zero evaluation metric scores, when the source dataset is IIIT-AR-13K. One possible reason could be that the financial documents of the source dataset vary vastly from the layout present in the target. In case of taking the source as PubLayNet, due to a closer alignment with medical research documents as the source, the metric scores are much higher than in case of IIIT-AR-13K. Still the scores are quite low compared to the promising results they displayed during the training phase and ultimately suffer from the domain shift.

However, fine tuning the models on additional synthetic noisy data in the form of RanLayNet shows drastic improvement across all the domains of the target dataset. The results display that fine-tuning the source dataset with our noisy dataset significantly enhances model performance across various metrics in different target domains. This finding supports the effectiveness of our hypothesis to enhance adaptability and generalization capabilities of models by pretraining them on synthetic noisy labeled data. The improvements can be visualized through the radar charts in Figure 4. Some of the detections on the target Doclaynet dataset are shown in Figure 5.

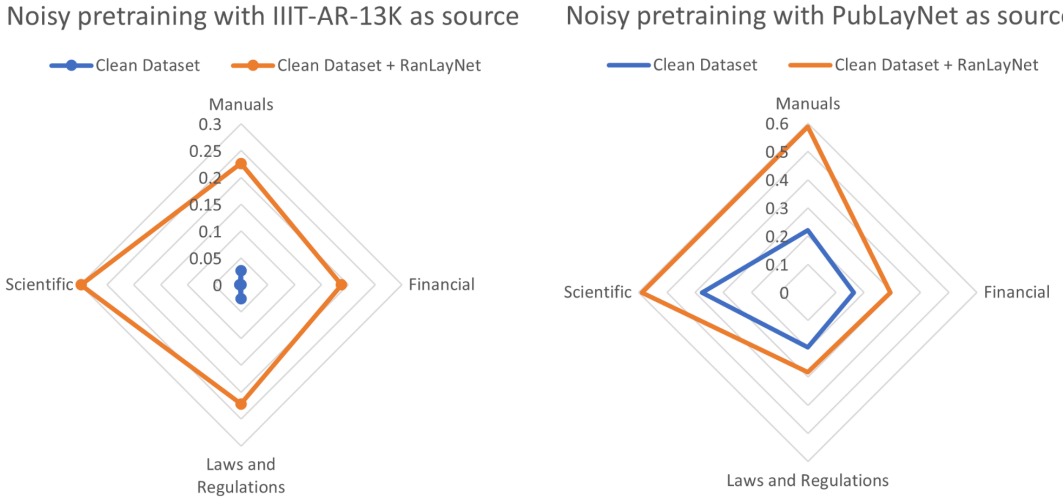


Figure 4 - Evaluation results of the test dataset before and after including Noisy Pretraining data at the source

Figure 5 - Visualization of inference results on the Doclaynet images after training detector on PubLayNet + RanLayNet

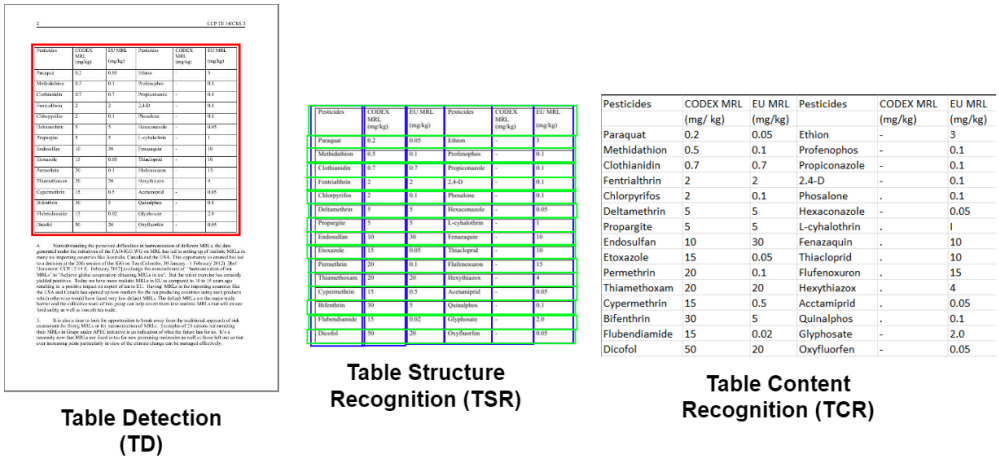
Chapter 2

Table Structure and Content Recognition (Table-OCR)

2.1 Table Recognition - Problem Statement and Challenges

The task of image based table recognition involves extracting the table structure and content from table crops from document images, and then to save it into a relevant representation format. To efficiently use data from table images, computer vision-based pattern recognition methods are used. *Table Detection (TD), Table Structure recognition (TSR), andTable Content Recognition (TCR)* are the three main tasks involved in *Table Recognition (TR)*. TD focuses on locating tables within images, TSR aims to recognize their internal structures, and TCR involves extracting the textual contents from the tables. The current emphasis is on developing end-to-end TR systems capable of seamlessly integrating all three sub-tasks. The primary goal is to address real-world scenarios where the system performs TD, TSR, and TCR simultaneously, thus enhancing the efficiency and effectiveness of table recognition in practical applications.
Thus, we can formulate the term ***Table OCR*****,** defined as

Table OCR = Table Detection (TD) + Table Structure Recognition (TSR) + Table Content Recognition (TCR)



Pesticides	CODEX MRL (mg/ kg)	EU MRL (mg/kg)	Pesticides	CODEX MRL (mg/kg)	EU MRL (mg/kg)
Paraquat	0.2	0.05	Ethion	-	3
Methidathion	0.5	0.1	Profenophos	-	0.1
Clothianidin	0.7	0.7	Propiconazole	-	0.1
Fentrialthrin	2	2	2,4-D	-	0.1
Chlorpyrifos	2	0.1	Phosalone	-	0.1
Deltamethrin	5	5	Hexaconazole	-	0.05
Propargite	5	5	L-cyhalothrin	-	I
Endosulfan	10	30	Fenazaquin	-	10
Etoxazole	15	0.05	Thiacloprid	-	10
Permethrin	20	0.1	Flufenoxuron	-	15
Thiamethoxam	20	20	Hexythiazox	-	4
Cypermethrin	15	0.5	Acctamiprid	-	0.05
Bifenthrin	30	5	Quinalphos	-	0.1
Flubendiamide	15	0.02	Glyphosate	-	2.0
Dicofol	50	20	Oxyfluorfen	-	0.05

Table Detection (TD)

Table Structure Recognition (TSR)

Table Content Recognition (TCR)

Figure 6 - The various stages of Table OCR - Table Detection (TD), Table Structure Recognition (TSR) and Table Content Recognition (TCR)

2.2 Related Works

2.2.1 TableTransformer

The Table Transformer (TATR) [8], first proposed in the PubTables-1M [9] dataset for table extraction and functional analysis, is a widely used model for TSR tasks. Despite the advancements in current open-source and commercial document analysis algorithms, such as the Table Transformer Model, certain limitations persist. For instance, affecting the model's ability to understand the context accurately. Additionally, when encountering tables with numerous empty cells or sparse content, the model might struggle to distinguish meaningful empty cells from those with missing data.

2.2.2 DEtection TRansformer

DEtection TRansformer (DETR) [10]**,** is built around key concepts like a set-based global loss that ensures unique predictions via bipartite matching and a transformer-based encoder-decoder architecture. The approach reframes object detection as a direct set prediction problem, minimizing the need for manually designed components such as anchor generation or non-maximum suppression, which typically require task-specific expertise. We employ DETR as an end-to-end, transformer-driven solution for object detection, which generates bounding boxes and class labels directly. This method ensures precise, non-overlapping predictions, effectively addressing issues with duplicate detections commonly seen in YOLO-based detectors.

2.2.3 CascadeTabNet

CascadeTabNet [12], a sophisticated end-to-end deep learning framework designed to effectively address both table recognition tasks within a unified model. This approach performs pixel-level table segmentation, accurately detecting each table instance in the input image, and also handles table cell segmentation by predicting segmented regions corresponding to individual cells, enabling the extraction of the table's structure. The model generates cell region predictions in a single inference pass. Furthermore, it classifies tables into two categories: bordered (ruling-based) and borderless (no ruling-based) tables. For borderless tables, the model directly predicts cell segmentation. The architecture integrates key components such as the Cascade RCNN [13], a multi-stage model tailored for high-quality object detection in convolutional neural networks (CNNs). It also incorporates a modified HRNet [14], which provides robust high-resolution representations and multi-level features that are advantageous for the semantic segmentation tasks involved in table recognition.

2.2.4 PPOCR-v2

The PP OCRv2 [15] is a reliable off the shelf system designed to achieve high accuracy and computational efficiency for practical OCR applications. For text detection, the proposed Collaborative Mutual Learning (CML) and Copy Paste are two methods for data enhancement that have been successful in improving accuracy for object detection and instance segmentation tasks. CML involves training two student networks and a teacher network to develop a more robust text detector, and it also proves to be beneficial for text detection. Moreover, in text recognition, they introduced the Lightweight CPU Network (PP-LCNet), Unified-Deep Mutual Learning (U-DML),and CenterLoss. U-DML makes use of two student networks to improve text recognition precision. CenterLoss helps reduce errors caused by comparable characters.

2.3 Methodology

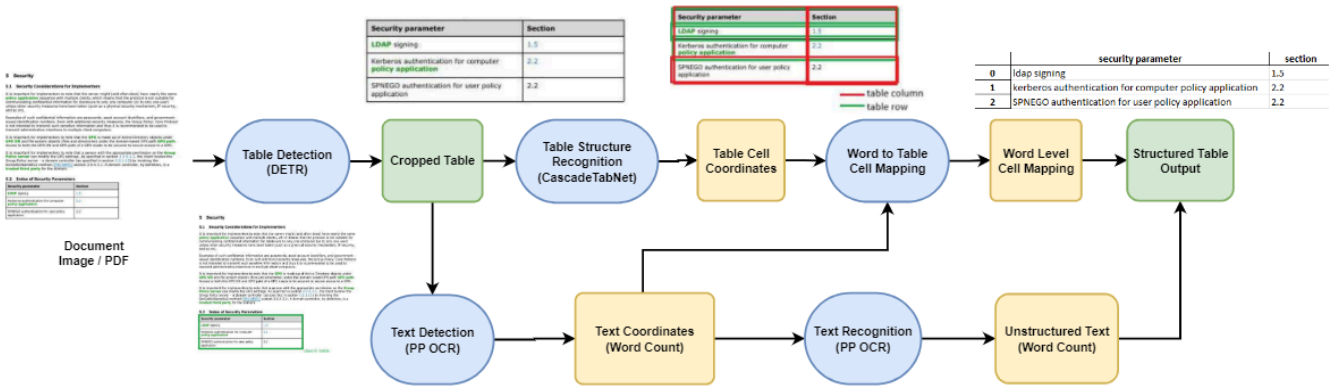


Figure 7 - Overall pipeline of TC-OCR for performing end-to-end Table OCR

We follow a comprehensive, end-to-end pipeline for table recognition by integrating a light, yet state of the art CascadeTabNet for TSR, a powerful detection model called DETR and off the shelf PP-OCR-v2 and compare its effectiveness against the Table Transformer model. We will later use this pipeline as an application in our later studies in Chapter 3.

Our proposed pipeline, **TableCraft-OCR (TC-OCR)** is visualized in Figureure 7 **.** It consists of the following stages -

2.3.1 Table Detection (TD)

We have selected DEtection TRansformer (DETR) for detecting table crops in document images. This module takes the document image as input and returns the table crop as an output, which is fed further into our pipeline.

2.3.2 Table Structure Recognition (TSR)

We have employed CascadeTabNet to fetch the location of rows and columns in the table. Given a table crop as input, it returns the bounding boxes of the table rows and columns. These boxes are used later in the pipeline for word to table cell mapping

2.3.3 Table Content Recognition (TCR)

For table content detection, we use a standard OCR library. OCR consists of two stages - *text detection* and *text recognition*.
In text detection, it finds the bounding boxes for each text word present in the image. Following this, each text crop is sent to the text recognition module, which generates the text present in the crop. These two stages build a standard OCR system. In our case, we have used PPOCR-v2.
The coordinates of the text crops from the intermediate text detection phase parallelly sent to the Word-to-Table-Cell mapping module along with the table structure information for reconstructing the complete table.

2.3.4 Word to Table Cell Mapping (WTCM)

The Table Cell Mapping module links each detected word from the OCR to a particular cell in the table structure.
It consists of three main steps -
Step 1 -

Calculate the centroid of each table cell (i,j) based on the bounding box coordinates present in the *xy-xy format* (top-left-bottom-right). TCN_{ij} represents the coordinates for the centroid of table cell at location (i,j)

$$\mathcal{TCN}_{i,j} = \left(\frac{\mathcal{TC}_{i,j}(x_1) + \mathcal{TC}_{i,j}(x_2)}{2}, \frac{\mathcal{TC}_{i,j}(y_1)) + \mathcal{TC}_{i,j}(y_2)}{2} \right) \text{— (Equation 1)}$$

Step 2 -

Similarly calculate the centroid of each detected word k based on the *xy-xy* bounding box. ECN_k represents the centroid for detected word k.

$$\mathcal{ECN}_k = \left(\frac{\mathcal{EC}_k(x_1) + \mathcal{EC}_k(x_2)}{2}, \frac{\mathcal{EC}_k(y_1)) + \mathcal{EC}_k(y_2)}{2} \right)$$ — (Equation 2)

Step 3 -

Finally we map each word k to a cell at row i and column j using simple euclidean distance. A flexible threshold, set at half the cell's width and length, accommodates variations in word positioning. This approach preserves empty cells and avoids incorrect mappings, preventing text misalignment and enhancing word-to-cell precision.

$$\mathcal{EC}_k = \begin{cases} (\mathcal{R}_i, C_j), & if |\mathcal{EN}_k(x) - \mathcal{TN}_{i,j}(x)| \leq \frac{\mathcal{W}(\mathcal{T}_{i,j})}{2} \\ & AND \\ & |\mathcal{EN}_k(y) - \mathcal{TN}_{i,j}(y)| \leq \frac{\mathcal{W}(\mathcal{T}_{i,j})}{2} \\ \phi, & \text{otherwise} \end{cases}$$ — (Equation 3)

2.4 Experimentation

We benchmark two simultaneous pipelines built using DETR, TableTransformer, CascadeTabNet and PPOCR-v2. We keep the same detection model and OCR system, while varying the TCR module. We call these two systems TATR and TC-OCR respectively.

TATR : DETR + TableTransformer + PPOCR-v2
TC-OCR : DETR + CascadeTabNet + PPOCR-v2

We run both of these pipelines on 240 manually annotated table crops taken from the TableBank [16] dataset consisting of two, three and four column tables. The inference time and row-level OCR accuracy are calculated for evaluation purposes. The inferences were made on a RTX A6000 GPU.

2.5 Results and Discussion

Columns	Rows	Total Images	TATR	TC-OCR	TATR Row Accuracy	TC-OCR Row Accuracy
2	1130	100	838	1075	*74.16*	***95.13***
3	1085	100	760	1010	*70.04*	***93.08***
4	570	40	220	400	*38.59*	***70.17***
Total	2785	240	1818	2485	*65.27*	***89.22***

Table 6 - OCR accuracy for TC-OCR and TATR pipelines on images taken from TableBank [16]

Model	Inference Time (seconds)	
TATR	Max	*15.48*
	Min	*4.95*
	Avg	*12.43*
TC-OCR	Max	***12.7***
	Min	***5.42***
	Avg	***8.23***

Table 7 - Statistics for inference time of 240 manually assisted crops TableBank[16] crops

Empirical results in Table 6 show that using CascadeTabNet as a backbone for TSR gives a higher boost in the overall table recognition task. As the table size increases, performance begins to drop sharply. TableTransformers, being generative models, will struggle to find the row and column coordinates as the table size increases. Due to the computational complexity and maximum sequence length constraint of Transformers, capturing long-range dependencies between cells can be challenging. As a result, lengthy tables suffer from information loss. This messes up the word-to-table-cell mapping algorithm and lowers down the row level accuracy.

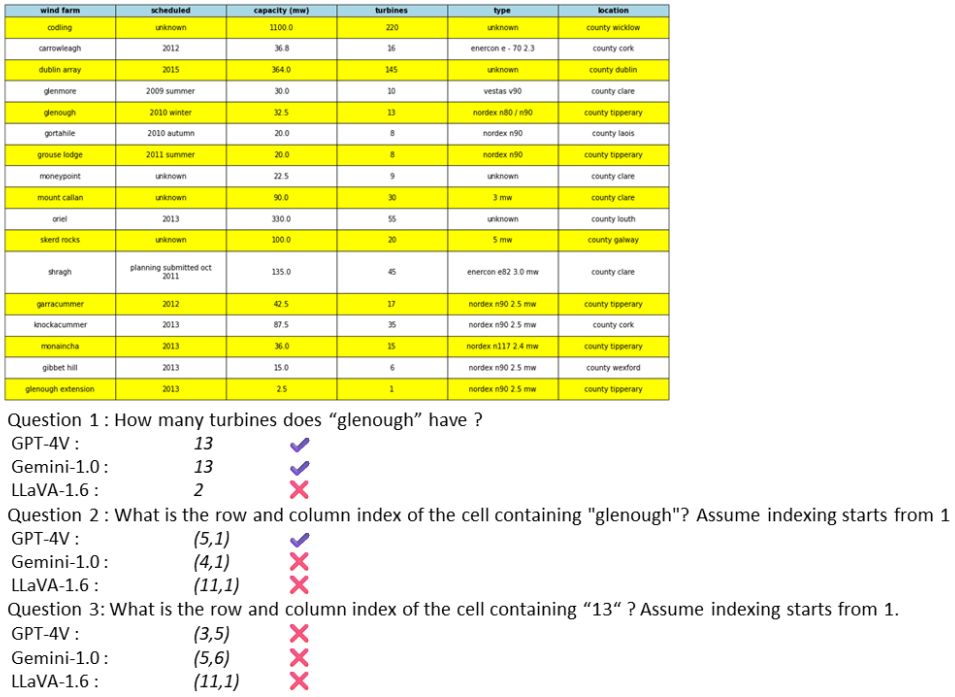
CascadeTabNet is immune to this problem as it solves the TSR task through segmentation and not direct bounding box generation. Moreover, the CascadeTabNet pipeline has faster inference time (30 % faster as seen in Table 7) on average and is suitable for real time Table Recognition tasks.

We will use the TC-OCR pipeline to provide the table representation in textual modality format in the Chapter 3.

Chapter 3

Table Visual Question Answering

3.1 - Table VQA and its underlying problem



wind farm	scheduled	capacity (mw)	turbines	type	location
codling	unknown	1100.0	220	unknown	county wicklow
carrowleagh	2012	36.8	16	enercon e - 70 2.3	county cork
dublin array	2015	364.0	145	unknown	county dublin
glenmore	2009 summer	30.0	10	vestas v90	county clare
glenough	2010 winter	32.5	13	nordex n80 / n90	county tipperary
gortahile	2010 autumn	20.0	8	nordex n90	county laois
grouse lodge	2011 summer	20.0	8	nordex n90	county tipperary
moneypoint	unknown	22.5	9	unknown	county clare
mount callan	unknown	90.0	30	3 mw	county clare
oriel	2013	330.0	55	unknown	county louth
skerd rocks	unknown	100.0	20	5 mw	county galway
shragh	planning submitted oct 2011	135.0	45	enercon e82 3.0 mw	county clare
garracummer	2012	42.5	17	nordex n90 2.5 mw	county tipperary
knockacummer	2013	87.5	35	nordex n90 2.5 mw	county cork
monaincha	2013	36.0	15	nordex n117 2.4 mw	county tipperary
gibbet hill	2013	15.0	6	nordex n90 2.5 mw	county wexford
glenough extension	2013	2.5	1	nordex n90 2.5 mw	county tipperary

Figure 8 - Inference of Table VQA on TabFact [17] dataset with GPT-4V, Gemini-Pro-1.0 and LLaVA-1.6. The underlying problem relates to poor Table Structure Understanding.

Question Answering (QA) is a classical way of evaluating the performance of a Large Language Models(VLMs). VQA (Visual Question Answering) tasks have become quite popular with the introduction of Large Visual Language Models (VLMs). Table Question Answering is no exception. There has been development in high quality Table QA and VQA datasets to test the table interpretability of LLMs and VLMs in recent times. Datasets like *TabFact [17], WikiTableQuestions [18], FeTaQA [19]* are widely used for Table QA tasks. The level of granularity increases from simple fact verification to one word answers and finally to descriptive answers from the table content. Domain specific models have also been trained end-to-end on table QA tasks like *Table-LLaMA [20]* and *Table-LLaVA [21].* Large closed source models like Gemini-Pro and GPT-4 also have acquired decent table reasoning capabilities over time.

However, we found that even the large scale models suffer from the structure interpretation. Even after giving a correct response from the content, LLMs and VLMs cannot localize the answer present within the table content. We have tried this on GPT-4V [29], GeminiPro1.0 [28] and LLaVA-1.6 [1] . This is shown in Figure 8 **.** Large closed source models can perform Table VQA to some extent, but mostly fail on table structure question answering, while open source models like LLaVA[1] don't work at all for reasonably sized tables.

3.2 - Current Benchmark Datasets for Table Structure Understanding

Comprehensive table structure understanding datasets have been quite limited. SUC Benchmark [22] is probably the first proper benchmark dataset released to test the capabilities of LLMs on textual tables. The SUC benchmark is built upon tables extracted from TabFact [17], ToTTo [23], HybridQA [24], SQA [25] and Feverous [26] table question answering datasets. It was built to investigate the best textual representation for table serialization and evaluate the existing table understanding capabilities of LLMs. Although it comes with a good set of tasks, they are fairly simple and not comprehensive enough to fully test the extent of table interpretability of LLMs. The block diagram of the benchmark tasks are visualized in Figure 9.

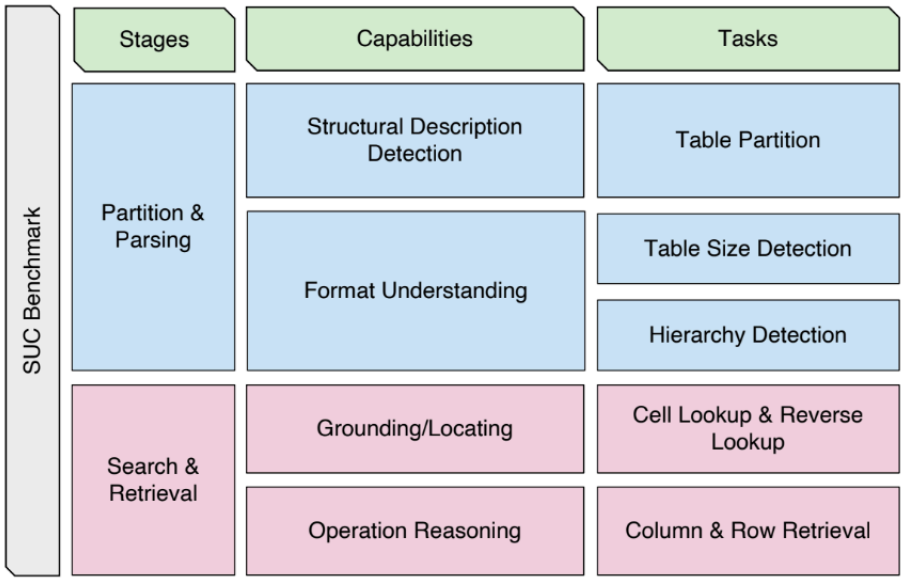


Figure 9 - Block diagram of the test suite of SUC Benchmark [22]

MMTab [21] is another dataset that included a few Table Structure Understanding tasks which was used as a pre-training objective for training TableLLaVA. It was designed to test the visual table interpretation capabilities of vision-language models. It introduced

table level augmentations by rendering table images in web, excel and markdown formats. It contains 8000 table images and six corresponding table structure understanding tasks - table size detection, table cell extraction, merged cell detection, row and column extraction,table cell location, row-column extraction and table recognition.

3.3 - TabBench - A new Benchmark Dataset

We propose ***TabBench*** - a new comprehensive benchmark dataset for evaluating the Table Understanding Capabilities of LLMs and VLMs. It is built upon the existing datasets, with the introduction of *ranged retrieval*, *two-dimensional sub-table retrieval* tasks and a couple of *table modification* tasks.

The TabBench benchmark introduces 13 table structure understanding tasks across 5 levels of perception. This is visualized in the block diagram shown below in Figure 10.

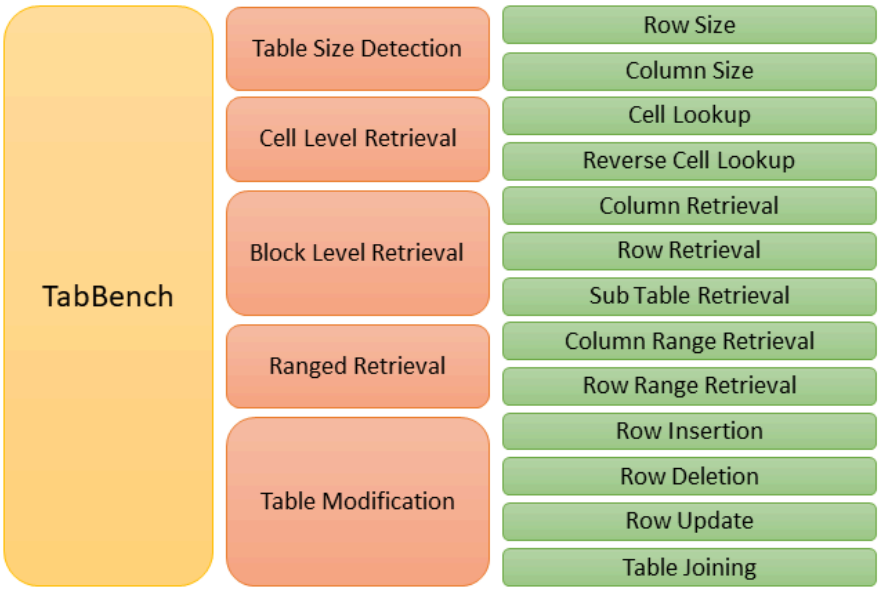


Figure 10 - Block diagram of the test suite of our TabBench benchmark

The tasks can be categorized into 5 levels of operations required in increasing order of complexity from cell level search to block level search, ranged retrieval and table

structure modification. The definition of each task and the evaluation metric for the expected output is discussed in Table 8.

Task Name	Acronym	Definition	Evaluation Metric
Row Size	RS	Count the number of rows in the table	Accuracy
Column Size	CS	Count the number of columns in the table	
Cell Lookup	CL	Find the position of a particular cell element	
Reverse Cell Lookup	RCL	Find the cell element at a particular position	
Column Retrieval	CR	Return the column contents at a particular column index	Cell wise accuracy
Row Retrieval	RR	Return the row contents at a particular row index	
Sub Table Retrieval	STR	Return the sub-table formed by slicing at a particular range of indices	TEDS
Column Range Retrieval	CRR	Return the minimum and maximum value at a particular column	Accuracy
Row Range Retrieval	RRR	Return the minimum and maximum value at a particular row	
Row Insertion	RI	Insert a new row into the table at a particular index and return the modified table	TEDS
Row Deletion	RD	Delete a row from the table at a particular index and return the modified table	

Table 8 - List of benchmarks used in the comprehensive TabBench suite.

Input tables were taken from the SUC [22] benchmark dataset and the ground truths were generated synthetically using specially handcrafted python scripts. The SUC

benchmark was designed to test LLMs and not VLMs. Therefore, python libraries were used to render the text-based table representations into high-quality tabular images. Outlier tables having very low or high row/column count were filtered off and the resulting dataset consists of ***3337 tables*** sampled from TabFact [17], ToTTo [23], HybridQA [24], SQA [25] and Feverous [26] Table QA datasets preset in the SUC benchmark. This dataset statistics for each task is shown in Table 9.

Source dataset	**TabFact**	**ToTTo**	**HybridQA**	**SQA**	**Feverous**	**Total**
Table count	822	540	546	781	648	3337

Table 9 - Table data statistics taken for TabBench from source datasets.

3.4 - LLM Evaluation

3.4.1 - Table Formats

event name	established	category	sub category	main venue
dieppe kite international	2001	sporting	kite flying	dover park
the frye festival	2000	arts	literary	university of moncton
hubcap comedy festival	2000	arts	comedy	various
touchdown atlantic	2010	sporting	football	moncton stadium
atlantic nationals automotive extravaganza	2000	transportation	automotive	moncton coliseum
world wine & food expo	1990	arts	food & drink	moncton coliseum
shediac lobster festival	1950	arts	food & drink	shediac festival grounds
mosaã¯q multicultural festival	2004	festival	multicultural	moncton city hall plaza

Figure 11 - A sample table taken from the TabBench Dataset

Each table is represented in 5 textual formats - html, markdown, xml, csv and json. The raw tables were present in json format. We have converted them to the other formats using python scripts. The tabular representations for the sample table shown in Figure 11 is as follows -

1) HTML

```
<html>
<body>
<table>
<thead>
<tr><td>event name</td><td>established</td><td>category</td><td>sub category</td><td>main
venue</td></tr>
</thead>
<tbody>
<tr><td>dieppe kite international</td><td>2001</td><td>sporting</td><td>kite
flying</td><td>dover park</td></tr>
<tr><td>the frye festival</td><td>2000</td><td>arts</td><td>literary</td><td>university of
moncton</td></tr>
<tr><td>hubcap comedy
festival</td><td>2000</td><td>arts</td><td>comedy</td><td>various</td></tr>
<tr><td>touchdown atlantic</td><td>2010</td><td>sporting</td><td>football</td><td>moncton
stadium</td></tr>
<tr><td>atlantic nationals automotive
extravaganza</td><td>2000</td><td>transportation</td><td>automotive</td><td>moncton
coliseum</td></tr>
<tr><td>world wine & food expo</td><td>1990</td><td>arts</td><td>food &
drink</td><td>moncton coliseum</td></tr>
<tr><td>shediac lobster festival</td><td>1950</td><td>arts</td><td>food &
drink</td><td>shediac festival grounds</td></tr>
<tr><td>mosaã¯q multicultural
festival</td><td>2004</td><td>festival</td><td>multicultural</td><td>moncton city hall
plaza</td></tr>
</tbody>
</table>
</body>
</html>
```

2) Markdown

```
| event name | established | category | sub category | main venue |
| --- | --- | --- | --- | --- |
| dieppe kite international | 2001 | sporting | kite flying | dover park |
| the frye festival | 2000 | arts | literary | university of moncton |
| hubcap comedy festival | 2000 | arts | comedy | various |
| touchdown atlantic | 2010 | sporting | football | moncton stadium |
| atlantic nationals automotive extravaganza | 2000 | transportation | automotive | moncton
coliseum |
| world wine & food expo | 1990 | arts | food & drink | moncton coliseum |
| shediac lobster festival | 1950 | arts | food & drink | shediac festival grounds |
| mosaã¯q multicultural festival | 2004 | festival | multicultural | moncton city hall plaza
|
```

3) XML

```
<table><header><column>event
name</column><column>established</column><column>category</column><column>sub
category</column><column>main venue</column></header><rows><row><cell>dieppe kite
international</cell><cell>2001</cell><cell>sporting</cell><cell>kite
flying</cell><cell>dover park</cell></row><row><cell>the frye
festival</cell><cell>2000</cell><cell>arts</cell><cell>literary</cell><cell>university of
moncton</cell></row><row><cell>hubcap comedy
festival</cell><cell>2000</cell><cell>arts</cell><cell>comedy</cell><cell>various</cell></ro
w><row><cell>touchdown
atlantic</cell><cell>2010</cell><cell>sporting</cell><cell>football</cell><cell>moncton
stadium</cell></row><row><cell>atlantic nationals automotive
extravaganza</cell><cell>2000</cell><cell>transportation</cell><cell>automotive</cell><cell>
moncton coliseum</cell></row><row><cell>world wine &amp;amp; food
expo</cell><cell>1990</cell><cell>arts</cell><cell>food &amp;amp; drink</cell><cell>moncton
coliseum</cell></row><row><cell>shediac lobster
festival</cell><cell>1950</cell><cell>arts</cell><cell>food &amp;amp; drink</cell><cell>shediac
festival grounds</cell></row><row><cell>mosaã¯q multicultural
festival</cell><cell>2004</cell><cell>festival</cell><cell>multicultural</cell><cell>moncton
city hall plaza</cell></row></rows></table>
```

4) CSV

```
event name,established,category,sub category,main venue
dieppe kite international,2001,sporting,kite flying,dover park
the frye festival,2000,arts,literary,university of moncton
hubcap comedy festival,2000,arts,comedy,various
touchdown atlantic,2010,sporting,football,moncton stadium
atlantic nationals automotive extravaganza,2000,transportation,automotive,moncton coliseum
world wine & food expo,1990,arts,food & drink,moncton coliseum
shediac lobster festival,1950,arts,food & drink,shediac festival grounds
mosaã¯q multicultural festival,2004,festival,multicultural,moncton city hall plaza
```

5) JSON

```
header :['event name', 'established', 'category', 'sub category', 'main venue']
rows:[['dieppe kite international', '2001', 'sporting', 'kite flying', 'dover park'], ['the
frye festival', '2000', 'arts', 'literary', 'university of moncton'], ['hubcap comedy
festival', '2000', 'arts', 'comedy', 'various'], ['touchdown atlantic', '2010', 'sporting',
'football', 'moncton stadium'], ['atlantic nationals automotive extravaganza', '2000',
'transportation', 'automotive', 'moncton coliseum'], ['world wine & food expo', '1990',
'arts', 'food & drink', 'moncton coliseum'], ['shediac lobster festival', '1950', 'arts',
'food & drink', 'shediac festival grounds'], ['mosaã¯q multicultural festival', '2004',
'festival', 'multicultural', 'moncton city hall plaza']]
```

3.4.2 - Prompt Design

The boiler plate prompt template for evaluating LLMs on TabBench is shown in Figure 12. It contains clear instructions about the problem statement, task name, task definition, input format, input table and the question with expected output format. Since LLMs have shown good results on unseen tasks with in-context learning, we introduce 1-shot prompts for all the tasks except the table-size detection task. We explicitly tell the LLM that the tables are indexed from zero and the row contains the table header too.

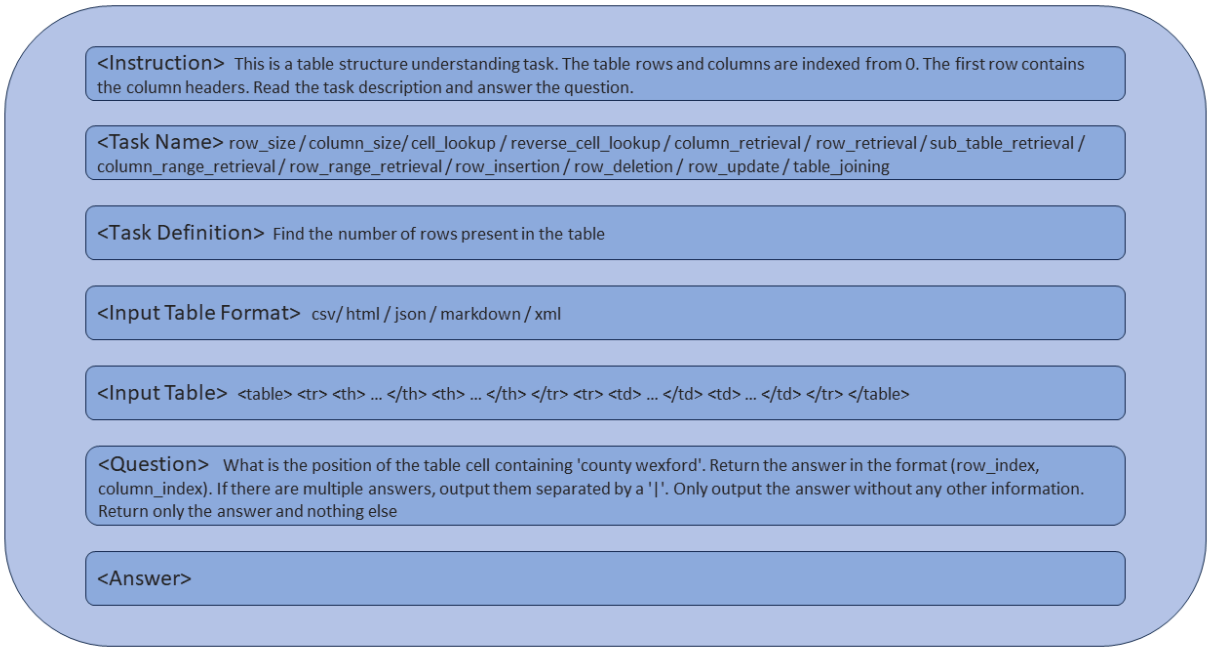


Figure 12 - Generalized prompt from the TabBench dataset

The detailed prompt for each task of TabBench is provided in Appendix A1.

3.4.3 - Experimentation

The TabBench suite is evaluated on both open source and closed source LLMs. We choose LLaMA-3 8B [27] parameter model and Gemini-Pro-1.0 [28]. The inferences were done on a A100 GPU for LLaMA-3 and Google Cloud API for Gemini. Default parameters were chosen for Gemini while for LLaMA-3 the inference was done by setting *max_new_tokens = 2048* (since some tasks require table generation as output), *temperature = 0.2* and *top_p = 0.9*. We run the tests on all the 13 structure understanding tasks across the five input formats and record the responses. For

evaluation, the responses were filtered using post-processing scripts and the respective metrics were calculated.

3.4.4 - Results and Discussion

Table Format	**Table Structure**		**Cell Level Search**		**Block Level Search**		
	RS	CS	CL	RCL	CR	RR	SRT
CSV	20.09 17.62	68.884 90.582	8.228 42.972	25.05 32.956	72.476 89.696	1.454 0.134	50.558 N/A
JSON	6.692 17.36	84.646 96.86	9.84 40.64	30.45 30.584	75.236 92.342	1.328 0.12	54.77 N/A
HTML	***34.802*** ***50.17***	***90.176*** 96.586	***13.472*** ***54.75***	***38.574*** ***36.95***	***78.922*** 93.31`	2.34 0.132	***63.256*** N/A
Markdown	17.724 9.186	73.168 84.838	10.27 45.028	28.424 31.832	58.614 91.188	1.188 0.122	60.744 N/A
XML	12.496 15.848	76.77 ***98.174***	12.702 43.68	29.408 35.458	65.568 ***94.348***	**2.366** **0.142**	60.276 N/A

Table 10 - Table Structure, Cell and Block Level Search metrics with LLaMA-3-8B (first entry) and Gemini-Pro-1.0 (second entry) models on TabBench

Table Format	**Ranged Retrieval**				**Table Modification**		
	CRR		RRR		RI	RD	RU
	Min	Max	Min	Max			
CSV	78.552 93.956	89.328 94.648	21.854 25.83	29.018	53.65	53.55	51.25
JSON	***82.486*** 94.39	91.834 95.128	22.852 27.408	30.838 24.708	55.87	54.92	52.63
HTML	80.12 ***96.332***	***93.824*** ***96.356***	***23.644*** ***27.866***	***44.192*** ***24.304***	***64.36***	***63.85***	***59.46***
Markdown	77.29 95.156	88.894 95.412	22.94 25.708	31.314 25.958 24.722	61.82	62.92	57.22
XML	75.044 96.07	91.732 96.192	22.524 27.67	28.26 24.654	60.98	61.21	55.93

Table 11 - Ranged Retrieval and Table Modification metrics with LLaMA-3-8B (first entry) and Gemini-Pro-1.0 (second entry) models on TabBench

In Tables 10 and 11, we report the corresponding metric scores for each test in the proposed TabBench suite. Each cell represents the scores by LLaMA and Gemini-Pro respectively.

It is interesting to see that tree-type representations- HTML and XML which start and end tags, perform significantly better than other representations for taks like Cell Level and Block Level Search. In Table Size, the Row Size metric is way higher for HTML, probably due to the presence of the <tr> tags, which explicitly indicate the beginning of the new line in the table structure. Same goes for columns, denoted by <td> or <tr> tags.

In Cell search and Reverse Cell Search, the same trend is observed.Block Level, being a 2D search, is a very good metric to test the localization capability of LLMs while understanding tabular Data.

Table Modification tasks, just like Sub Table Retrieval, is a demanding task, and shows superior performance for html representation.

The Radar plots in Figure 13 visualizes the increase in metric scores for all the table representations, across all the five subset- datasets of TabBench.

In Figure 14, we plot the microablation study, where each metric's running average is plotted as a function of table size. Table size is the total number of cells in the table. It is interesting to observe that HTML is very stable, and the running average drops very slowly as the table size increases.

From all these above observations, we conclude that HTML is the best format for interpreting tabular data by LLMs,probably due to the huge corpus of internet data it is trained on. This will hold true even for the language model inside VLMs, so we will choose HTML the primary table format for all purposes

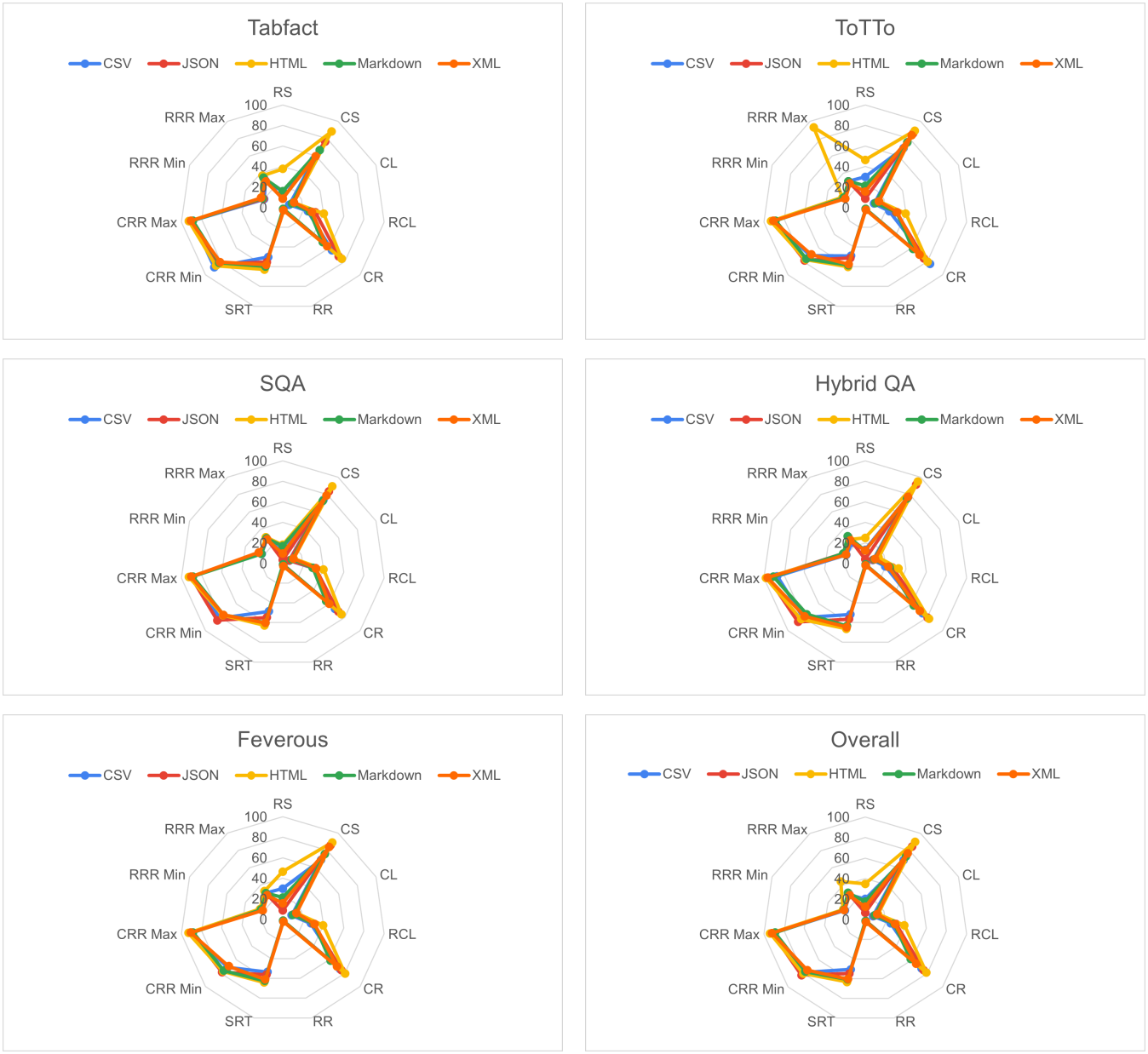


Figure 13 - Radar plot for metric scores of TabBench for LLaMA-3-8B across all the five subset-datasets, and the overall metric scores.

3.4.5 - Micro-ablations

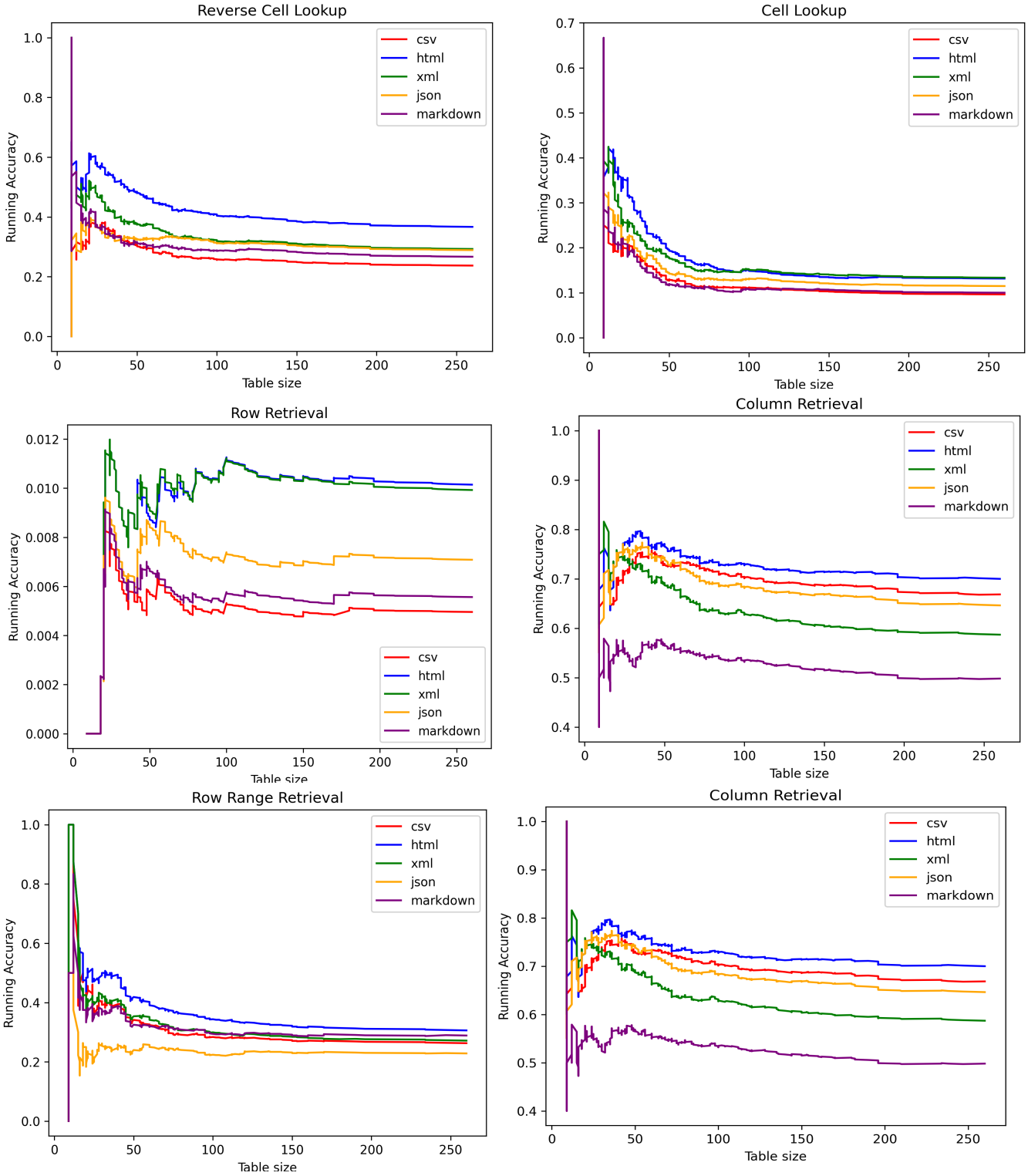


Figure 14 (a) - Metric's running average as plotted as a function of table size

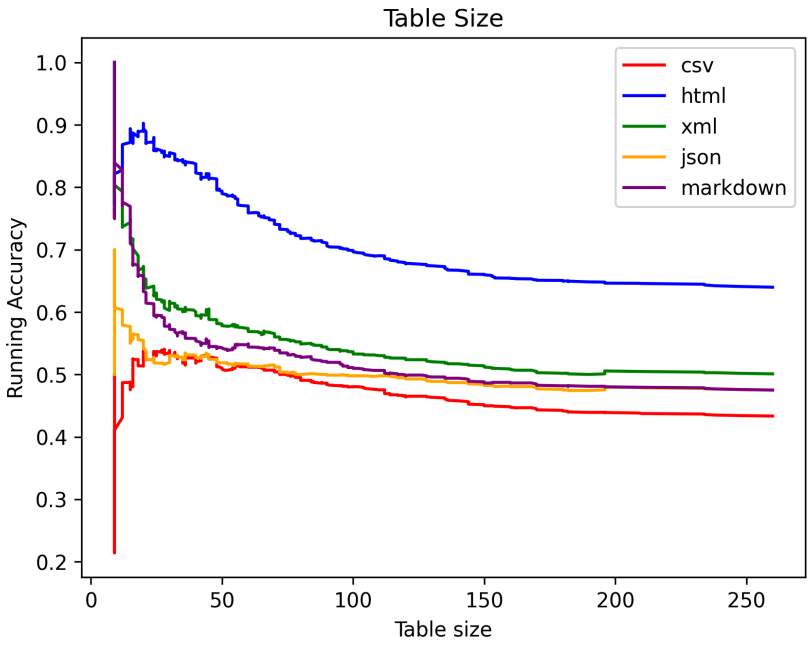


Figure 14 (b) - Metric's running average as plotted as a function of table size (Continued)

3.5 - Unimodal VLM Evaluation

Unimodal VLM Evaluation refers to Table VQA capabilities of VLMs, where the table is provided in the image modality, and the question/user query in the text modality.

3.5.1 - Experimentation

We use the same prompts from TabBench, minus the input table in the text format. The table is rendered as an image using the *html2image* package of python. Tables are rendered in three formats - borderless, bordered and colored (alternating row colors). A sample table is shown in Figure 11. We initially tried to use Gemini-1.0-Vision and LLaVA for evaluation. But LLaVA appears to not adhere to the output format, and is hence dropped due to difficulty to evaluate the metric scores from the un-ordered output responses.

We drop the Table Modification Tasks for Gemini due to expensive and time consuming API calls during response generation.

3.5.2 - Results and Discussion

The evaluation metrics for unimodal VLM benchmarks is shown in Table 12.

Table Format	Table Structure		Cell Level Search		Block Level Search			Ranged Retrieval			
	RS	CS	CL	RCL	CR	RR	SRT	CRR		RRR	
								Min	Max	Min	Max
Borderless	7.856	53.294	12.558	7.964	61.226	0.062	0.342	86.4	85.958	23.226	21.444
Bordered	7.974	57.83	12.642	8.282	62.858	0.07	0.354	87.276	87.412	22.902	21.092
Colored	8.634	60.568	12.846	8.366	63.92	0.082	0.353	86.914	87.258	22.90	21.918

Table 12 - TabBench evaluation metric scores for borderless, bordered and colored tables on Gemini-Pro-1.0-Vision

As expected, unimodal VLM Table VQA shows subpar performance in all the tasks, compared to their LLM counterparts (Table 11). The image modality is a bottleneck for VLMs while interpreting tabular image data, which drops table reasoning performance. Some of the reasons for this behaviour could be due to -

- Poor Image-Text alignment
- Visual Tokens are worse than native text tokens

In this context, we also test the feasibility of **Multimodal VLM Table VQA.** The table is provided both as an image and text (html format) to the VLM. The hunch is that the VLM should refer to the table image to comprehend the spatial arrangement and hierarchical structure, while relying on the text input for OCR and cross verification. In real world scenarios, the text can be extracted using pipelines like TC-OCR (Chapter 2). When carried out on Gemini-Pro-1.0-Vision, there is a bit of improvement, but it is nowhere close to the unimodal LLM metrics. Thus, even zero-shot multimodal Table VQA is not feasible, even for large models like Gemini. In Figure 15, we plot the running accuracy for Cell Lookup using Text, Image and Text + Image modalities with Gemini.

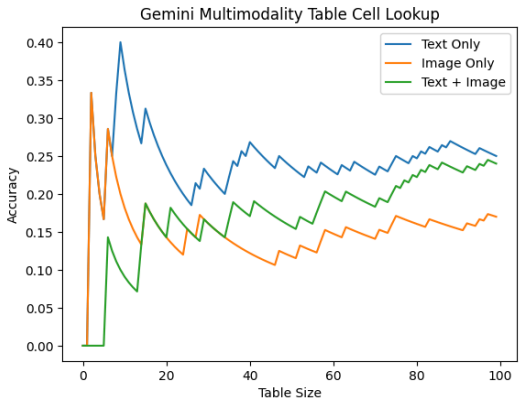


Figure 15 - Gemini Table VQA with multiple table modalities, for Cell Lookup task from TabBench

3.6 - The underlying problem with VLMs and related work

We can see a very significant drop in evaluation metric scores for all the table structure understanding tasks for both closed source and open source vision-language models. This poses a great concern, because in real-world scenarios, the tables are in the form of raw images, and cannot expect them to be available in processed text format. Therefore improving the table structure and understanding capabilities of VLMs is of great importance.

Recent studies have shown that VLMs lose their spatial understanding capabilities during their training process and perform very poorly on localisation tasks because the multimodal data contains mostly captions without explicit spatial grounding. In [30], the authors present an input-agnostic Positional Insert (PIN), a learnable spatial prompt with a minimal parameter set that integrates seamlessly within the frozen Vision-Language Model (VLM), enabling object localization and enhancing object detection capabilities in models such as BLIP-2 [31] and OpenFlamingo [32]. The PIN module is trained using a straightforward next-token prediction task on synthetic data, without the need for additional output heads. The architecture of their pipeline is visualized in Figure 16.

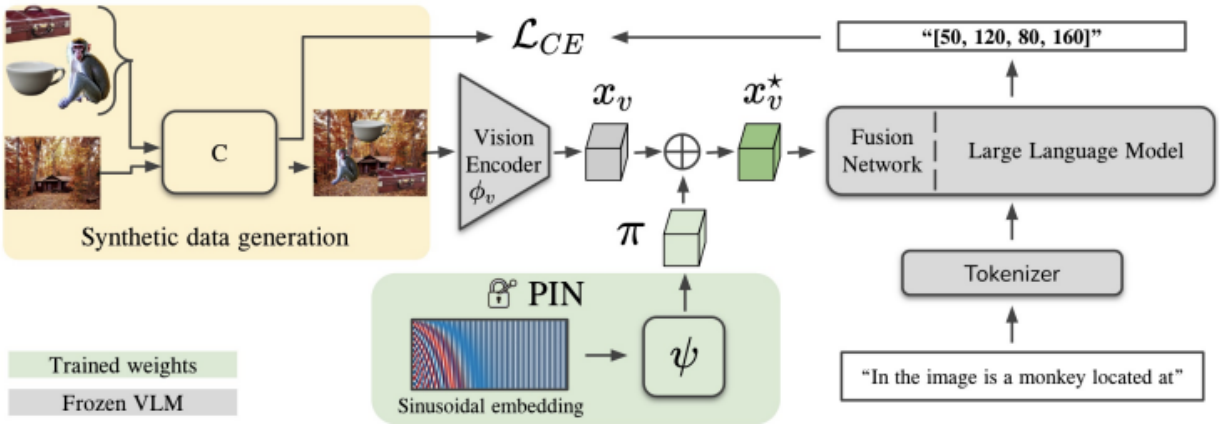


Figure 16 - Overall architecture of PIN module based VLMs

On a relevant note, it should also be noted that a Video-QA test done on SOTA VLMs [33] reveals that the language component overpowers the vision component, and can even generate correct answers based on the text input and the huge semantic knowledge present inside the language models. The vision-modality provided to the VLMs are not properly utilized in visual reasoning tasks and even blind-shot tests with empty video frames generate somewhat correct results. This is visualized in Figure 17.

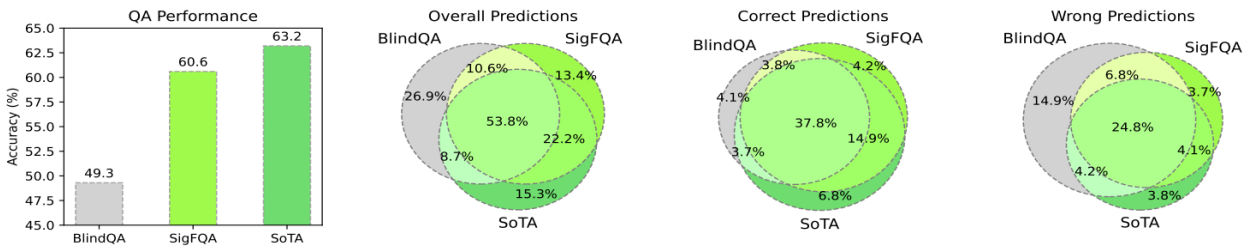


Figure 17 - Blind shot VQA done on VLMs.

In the recent InternVL [34,35] series of VLMs, the authors scale up the vision foundation model up to 6 billion parameters and progressively aligns it with a language model using web-scale image-text data from diverse sources. This achieves state-of-the-art performance across 32 generic visual-linguistic benchmarks, covering a range of tasks such as image-level and pixel-level recognition, zero-shot image/video classification, zero-shot image/video-text retrieval, and integration with language models to create multimodal dialogue systems.

Table-LLaVA [21] is the first approach to introduce structure understanding tasks during the VLM pre-training and fine-tuning processes on table question answering. The MMTab dataset, consisting of MMTab-pretrain, MMTab-instruct and MMTab-eval, is currently the only large-scale dataset specialized for grounding table structure information into VLMs. It consists of 150K samples of pre-training, 232K of instruction tuning and 45K + 4K of held-in and held-out test samples respectively with 105K tables of diverse domains and representations like Excel, Webpage and Markdown. These tables were collected from 14 public table datasets of 8 domains covering 9 academic tasks. In addition to three levels of table reasoning, i.e. Table Question Answering (TQA), Table Fact Verification (TFV) and Table to Text (T2T) requiring identity extraction, yes/no and reasoning backed answers respectively, it includes 6 Table Structure Understanding (TSU) tasks namely - Table Size Detection (TSD), Table Cell Extraction (TCE), Table Cell Locating (TCL). Merged Cell Detection (MSD), Row and Column Extraction (RCE) and Table Recognition (TR). Such tasks have proven to improve the generalized table reasoning capabilities of LLaVA-1.5 [1] significantly when evaluated on academic tasks. The set of tasks is shown in Figure 18.

MMTab	Task Category	Task Name	Dataset	Table Style	Domain	Held-in	# Tables		# Samples		Avg. Length
							Train	Test	Train	Test	(input/output)
MMTab-instruct	Table Question Answering (TQA)	Flat TQA	WTQ (2015)	W	Wikipedia	Yes	1.6K	0.4K	17K	4K	45.9/10.4
		Free-form TQA	FeTaQA (2022)	W	Wikipedia	Yes	8K	2K	8K	2K	32.3/18.69
		Hierarchical TQA	HiTab (2022)	E	Wikipedia goverment reports	Yes	3K	0.5K	8K	1.5K	63.5/12.6
			AIT-QA (2021)	E	Airline	No	-	0.1K	-	0.5K	41.8/10.2
		Multi-choice TQA	TabMCQ (2016)	M	science exams	No	-	0.05K	-	1K	47.9/13.2
		Tabular Numerical Reasoning	TABMWP (2023b)	W	math exams	Yes	30K	7K	30K	7K	54.2/51.9
			TAT-QA (2021)	M	financial reports	Yes	1.7K	0.2K	5.9K	0.7K	40.1/16.5
	Table Fact Verification (TFV)	TFV	TabFact (2020)	E, M	Wikipedia	Yes	9K	1K	31K	6.8K	49.9/18.3
			InfoTabs (2020)	W	Wikipedia	Yes	1.9K	0.6K	18K	5.4K	54.2/18.6
			PubHealthTab (2022)	W	public health	No	-	0.3K	-	1.9K	71.9/18.4
	Table to Text (T2T)	Cell Description	ToTTo (2020)	W	Wikipedia	Yes	15K	7.7K	15K	7.7K	31.1/14.8
			HiTab_T2T (2022)	E	Wikipedia goverment reports	Yes	3K	1.5K	3K	1.5K	39.1/14.7
		Game Summary	Rotowire (2017)	E	NBA games	Yes	3.4K	0.3K	3.4K	0.3K	27.6/291.7
		Biography Generation	WikiBIO (2016)	E	Wikipedia	Yes	4.9K	1K	4.9K	1K	18.1/84.2
	Table Structure Understanding (TSU)	Table Size Detection	TSD	W, E, M	-	Yes	8K	1.25K	8K	1.25K	30.1/17.9
		Table Cell Extraction	TCE	W, E, M	-	Yes	8K	1.25K	8K	1.25K	51.6/19.9
		Table Cell Locating	TCL	W, E, M	-	Yes	8K	1.25K	8K	1.25K	72.5/45.6
		Merged Cell Detection	MCD	W, E, M	-	Yes	8K	1K	8K	1K	57.49/28.2
		Row&Column Extraction	RCE	W, E, M	-	Yes	8K	1.25K	8K	1.25K	45.6/55.1
		Table Recognition	TR	W, E, M	-	Yes	8K	1K	8K	1K	16.3/389.2
	Total						82K	-	232K	-	66.1/66.9
MMTab-eval	Total						-	23K	-	49K	46.3/32.6
MMTab-pre	Table Recognition		TR	W, E, M	-	-	97K	-	150K	-	16.4/397.5

Figure 18 - Table of Tasks contained in the MMTab [21] splits

3.7 - Improving Table Reasoning capabilities of VLMs

Based on the observations in the related works, it is evident that work needs to be done to strengthen the vision component of VLMs to improve table reasoning in general.

At the dataset level, we can introduce better pre-training tasks to better align the image modality to the text counterpart. We have introduced an additional stage-2 pre-training to the MMTab training pipeline consisting of a novel **masked complementary table reconstruction task (MCTR).**

At the architecture level, we can introduce a **model agnostic positional insert (PIN) adapter,** inspired from the related works, to infuse spatial information into the image embeddings before feeding them as image tokens to the LLM.

We will also test if scaling up the VITs does any justice to remove the LLM bias, for our use case.

3.7.1 - Masked Complimentary Table Reconstruction

Traditional table reconstruction tasks included in datasets like MMTab require the model to reconstruct the text representation of the entire table in a format like *HTML* or *XML*. This kind of pre-training targeted for image-text alignment, provides rudimentary level of structure understanding capabilities because the VLM just has to read the data in a simple way from left to right, in a top down fashion. It just tests the OCR capabilities of the VLM

In order to incorporate strong spatial tabular understanding, we can modify the table reconstruction task as -

i) Split the table **T** into two complimentary, mutually exclusive and exhaustive tables $\mathbf{T_1}$ and $\mathbf{T_2}$ such that $\mathbf{T_1 \cup T_2 = T}$ and $\mathbf{T_1 \cap T_2 = \phi}$. We achieve this by masking out a cell in one table and preserving the cell in the other. In this way, we get two non-overlapping tables whose fusion recreates the original table.

ii) Render T_1 as an image and keep T_2 in text format like html. This way, we split the input data into both the text and image modalities, earlier which was just present in the image modality.

Thus the final pre-training objective is : $\mathbf{min\ (T_1^{image} \cup T_2^{text} - T)}$

By splitting the input table into two non-overlapping and complementary tables, and providing them as separate entries for each modality, should provide better image-text modality alignment compared to the vanilla table reconstruction objective. The pretraining task is visualized in Figure 19.

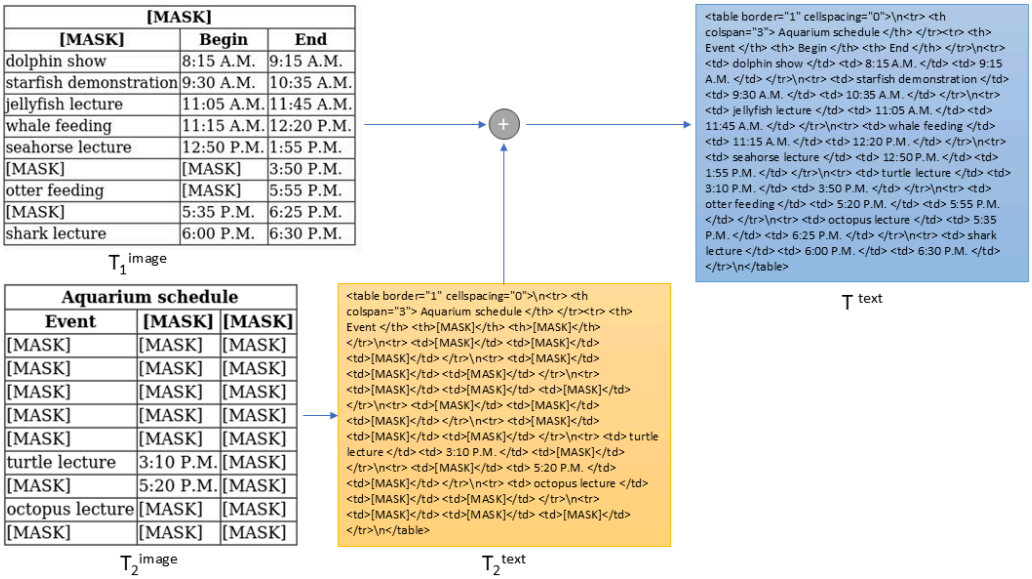


Figure 19 - The proposed Masked Complementary Table Reconstruction (MCTR) task.

3.7.2 - Positional Insert (PIN) Adapter

Positional Inserts are model agnostic spatial feature embeddings that are added to the output from the vision encoder to infuse spatial awareness to the vision embeddings. Positional embeddings are initialized with fixed sinusoidal embeddings that are fed to a learnable fully connected network as input. These embeddings are projected by the network to match the dimension of the output of the vision encoder.

If there are n vision tokens each of size d, then the positional embeddings $S^{n \times d}$ are initialized as -

$$s[i, 2k] = sin(\frac{i}{10000^{2k/d}}) \quad \text{— Equation 5}$$

$$s[i, 2k + 1] = cos(\frac{i}{10000^{2k/d}}) \quad \text{— Equation 6}$$

where i is the positional token ranging from 1 to n and k is the dimension of each embedding ranging from 1 to d. A sinusoidal embedding of shape 128 x 576 is shown in Figure 20.

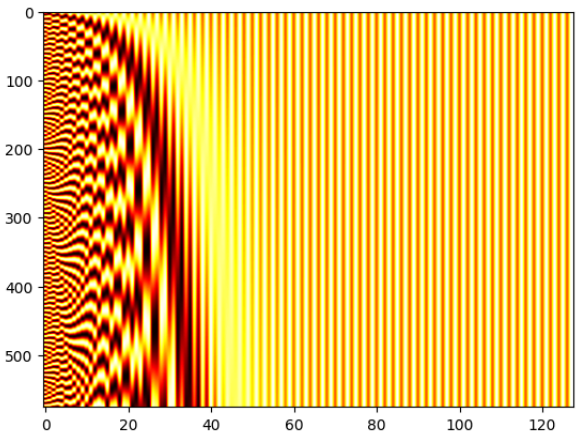


Figure 20 - Sample sinusoidal embedding of shape 128x576

The positional embeddings (s) are of the same dimensions as the image embeddings (x_v) generated by the vision encoder and hence can be added directly to generate the position-aware image embeddings (x_v^*).

$$x_v + s = x_v^* \quad \text{— Equation 7}$$

These position-enhanced image embeddings can be directly fed to the language model for generation purposes.

3.7.3 - Model Architecture and Training Strategy

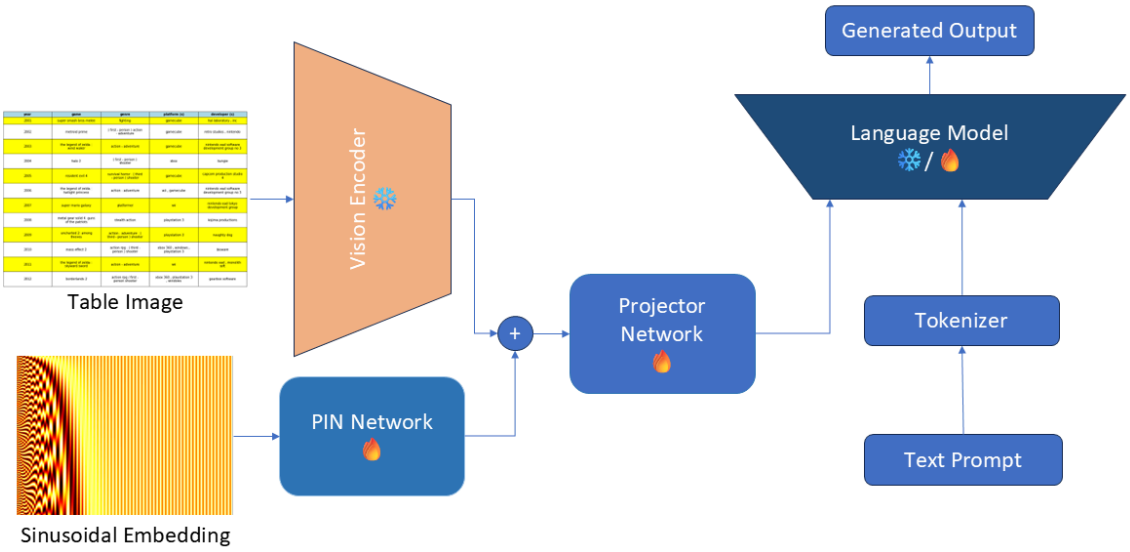


Figure 21 - overall architecture and training pipeline of our proposed VLM model.

Figure 21 shows the basic pipeline of the VLM incorporated with a Positional Insert (PIN) network. The Table image is fed to the vision encoder which generates 'n' number of image embeddings 'x_v', each of 'd' dimensions. The PIN network takes input sinusoidal embeddings generated by equations (5) and (6) and projects it to 'd' dimensional vectors 's'. Both the image embeddings and positional insert vectors are added element wise to generate spatial-aware image embeddings x_v^* The final vision embeddings are sent to a projector network that maps them to the same vector space as the text embeddings generated from the input text after passing through the tokenizer. FInally both the text and image tokens are fed to the language mode for generation/completion.

We deploy a two-stage pre-training followed by a single stage instruction-tuning on the MMTab enhanced with the complementary masked table reconstruction task. The vision encoder is always frozen, while the PIN and Projector networks are always always set as traininable. The LLM is trained only during the instruction-tuning stage. The overall training pipeline is summarized in Table 13.

Training Stage	Task	MM-Tab Dataset	Additional Dataset	Vision Encoder	PIN Network	Projector Network	LLM
Pretrain Stage-1	Table Reconstruction + Image Captioning	MM-Tab Pretrain (150K)	LLaVA-1.5 Pretrain (558K)	Frozen	Trainable	Trainable	Frozen
Pretrain Stage-2	Complimentary masked table reconstruction	MM-Tab Pretrain-Enhanced (130K)	LLaVA-1.5 Pretrain (558K)	Frozen	Trainable	Trainable	Frozen
Instruction -Tune	Image Captioning + VQA + Visual Grounding + Table Structure Task + Table VQA	MM-Tab Instruct (232K)	LLaVA-1.5-IFT (558K)	Frozen	Trainable	Trainable	Trainable

Table 13 - Various stages involved in training the Table VLM.

The training stages also contain additional tasks like image captioning, visual grounding, VQA etc from the original LLaVA training dataset to retain the generalized reasoning capabilities of the VLM.

3.7.4 - Evaluation Metrics

- For tasks like Table Question Answering (TQA), Table Fact Verification (TFT), we use **Accuracy** and **BLEU** scores.
- For Table Structure Understanding (TSU) Tasks, we use **row-wise and column-wise accuracy** for Table Size Detection (TSD), **cell level accuracy** for Table Cell Extraction (TCE) and Table Cell Location (TCL), **cell-level F1** for Merged Cell Detection (MCD), **cell-level F1** for row and column in case of Row and Column Extraction (RCE).
- For table generation tasks like Table Reconstruction (TR), Sub-Table Retrieval (SRT), Row Insert (RI), Row Deletion (RD) and Row Update (RU), we use **Tee-Edit-Distance-based-Similarity (TEDS)**, normalized between 0 and 1. For table-generation tasks involving Markdown or Latex, we convert the generated output to HTML and then compute the TEDS score.

3.7.5 - Baseline Models

We have chosen Table-LLaVA [21] and Intern-VL-chat-LLaVA [34,35] for our experimentations as both of them share the same LLaVA-1.5[1] architecture and

codebase. Both of them use the CLIP-VIT-L 336/14 and Intern-VIT vision encoders respectively while sharing a common Vicuna-1.5 [36] 7B LLM. The detailed architectural specifications of each variant is shown below in table 14.

Model	**Vision Encoder**	**Patch Resolution**	**Vision Encoder Embeddings/ PIN Embeddings size**	**LLM**
Table-LLaVA	CLIP-VIT-L 336/14	336 px	576x1024	Vicuna-1.5 7B
InternVL-1.0	Intern-VIT 300M	448px	1024x1024	Vicuna-1.5 7B
InternVL-1.0	Intern-VIT 6B V1.5	448px	1024x3200	Vicuna-1.5 7B

Table 14 - Baseline model architectures used for experimentations

3.7.6 - Experimentation

We train each model variant mentioned in Table 14 using the training pipeline mentioned in section 3.7.3. The hyperparameters and other specifications for each stage of the training pipeline are mentioned in detail in Table 15.

The experiments were done on systems having 4x Nvidia-A100-80GB and 4x Nvidia-A40-45GB GPUs, each with 512 GB of RAM. Batch sizes were adjusted to 4/8 accordingly for each pretrain/finetune stage to fit the GPU VRAM of each system.

Hyperparameter/ Specification	**Pretrain-Stage-1**	**Pretrain-Stage-2**	**Instruction-Fine Tune**
Primary Training Dataset	MMTab-pretrain (150K)	MMTab-pretrain-enhanced (130K)	MMTab-Instruct (232K)
Secondary Training Dataset	LLaVA-1.5-pretrain (558K)	LLaVA-1.5-pretrain (558K)	LLaVA-1.5-Instruct (665K)
Batch Size	32		16
Max Token Length	2560		
Learning Rate	1e-3		2e-5
LR Scheduler	Cosine decay		

Warmup Ratio	0.3	
Weight Decay	0	
Optimizer	AdamW	
Epochs	1	
Deepspeed Optimizer	Stage 2	Stage 3
Flash Attention	Level 2	

Table 15 - hyperparameters used during various training stages

3.7.7 - Results and Discussion

VLM	**PIN**	**TSD**	**TCE**	**TCL**	**MCD**	**RCE**	**TSD**	**TCE**	**TCL**	**RCE**
		Held-in Dataset					**Held-out Dataset**			
		Row/Col Acc	**Acc**	**Acc**	**F1**	**Row/Col F1**	**Row/Col Acc**	**Acc**	**Acc**	**Row/Col F1**
Table-LLaVA (baseline)	✗	0.460 0.755	0.309	0.452	0.411	0.376 0.516	0.368 0.500	0.241	0.364	0.223 0.267
Table-LLaVA (ours)	✓	**0.471** **0.764**	**0.315**	**0.456**	**0.378**	**0.381** **0.552**	0.352 **0.516**	**0.246**	**0.377**	**0.250** 0.263
InternVL (baseline)	✗	0.385 0.676	0.275	0.404	0.343	0.373 0.554	0.3 0.448	0.228	0.319	0.307 0.287
InternVL (ours)	✓	**0.441** **0.712**	**0.326**	**0.4496**	**0.346**	**0.395** **0.563**	**0.36** **0.54**	**0.249**	**0.382**	**0.313** **0.392**

Table 16 - Evaluation results on Table Structure Understanding tasks with models trained with and without the PIN adapter.

Table 16 shows the evaluation results of the Table Structure Understanding tasks on the MMTab-Test dataset both for held-in and held-out splits.These set of tasks test the localization and visual grounding capabilities of the VLMs for tabular data.

The baseline Table-LLaVA trained with a CLIP-VIT-L 336/14 vision encoder resembles metric scores quite similar to the original paper. The addition of the Positional-Insert (PIN) module has improved the baseline scores in all of the 5 held-in dataset tasks.

There seems to be a minor drop in the held-out datasets, in 2 out of 4 held-out datasets, in the row accuracy of TSD and column F1 of RCE task. This is probably due to a domain-shift in the held-out datasets, and the weaker CLIP encoder for domain-generalization in comprehending complex tabular data.
We observe the similar trend in the case of the InternVL model. The PIN module seems beneficial, which is evident from the increase in baseline scores for all the tasks, for both the held-in and held-out datasets. It is also interesting to note that, the vision encoder of Intern-VL, the Intern-VIT-300M is over 50% smaller in size than the corresponding CLIP-encoder of Table-LLaVA, yet it scores similar results in the held-in dataset. Moreover, it beats Table-LLaVA in all the held-out datasets, with a huge boost in the held-out RCE task metric. This is probably due to a combination of the higher resolution (448px vs 336px) and finer-grained visual tokens (1024 vs 576). Theoretically, the Intern-VIT should also have better OCR and visual grounding capabilities then the CLIP-encoder because it is a distilled version of the larger Intern-VIT-6B model, hence better is the generalization capability for TSU tasks during domain shifts.

VLM	**PIN**	**Table Reconstruction**			**Sub-Table retrieval**	**Row Insertion**	**Row Deletion**	**Row Update**
		MMTab-Test			**Tab-Bench**			
		HTML (TEDS)	**Markdown (TEDS)**	**Latex (TEDS)**	**HTML (TEDS)**	**HTML (TEDS)**	**HTML (TEDS)**	**HTML (TEDS)**
Table-LLaVA (baseline)	✗	0.632	0.658	0.639	0.883	0.912	0.899	0.907
Table -LLaVA (ours)	✓	**0.641**	**0.666**	**0.647**	**0.884**	**0.916**	**0.903**	**0.91**
InternVL (baseline)	✗	0.645	0.695	0.697	0.908	0..926	0.918	0.888
InternVL (ours)	✓	**0.668**	**0.716**	**0.711**	**0.911**	**0.928**	**0.921**	**0.923**

Table 17 - Evaluation results on Table Structure Generation tasks with models trained with and without the PIN adapter.

In Table 17, we evaluate the VLMs for their Table Generation capabilities, as a part of the TSU task. MMTab-Test tests for entire table-reconstruction, thus testing the end-to-end OCR capacity of these models. Tab-Bench tests the corresponding table-slicing and table-modification capabilities of such models.

As expected, the PIN module improves the baseline performance for both Table-LLaVA and Intern-VL for table reconstruction. The inclusion of this module has benefited the respective visual embeddings fed to the LLM by incorporating spatial information and thus improving the visual grounding capacity. The TEDS score for Intern-VL is once again way higher than the corresponding Table-LLaVA model due to the light weight, yet superior performance of the distilled Intern-VIT over the vanilla CLIP-encoder.

VLM	**PIN**	**Question Answering**					**Fact Verification**		**Text Generation**		
		Tab MWP (Acc)	**WTQ** (Acc)	**HiTab** (Acc)	**TAT-QA** (Acc)	**FeTa QA** (Bleu)	**Tab Fact** (Acc)	**Info Tabs** (Acc)	**Hi Tab-T2T** (Bleu)	**Roto wire** (Bleu)	**Wiki Bio** (Bleu)
Table-LLaVA (baseline)	✗	0.862	0.207	0.117	0.263	29.93	0.635	0.697	9.84	9.84	11.31
Table -LLaVA (ours)	✓	**0.866**	**0.216**	**0.128**	**0.272**	**30.48**	**0.64**	**0.704**	**9.93**	**10.93**	**11.48**
InternVL (baseline)	✗	0.842	0.184	0.117	0.307	25.52	0.601	0.65	8.86	11.09	7.54
InternVL (ours)	✓	**0.852**	**0.209**	**0.124**	**0.359**	**29.27**	**0.618**	**0.658**	**9.10**	10.83	**8.91**

Table 18 - Evaluation results on academic Table VQA datasets with and without the PIN adapter.

In table 18, we report the academic Table VQA scores for all the models with and without the PIN adapter. We see improvements in 19 out of 20 tasks, including Question Answering, Fact Verification and Text Generation. The PIN adapter enhances the spatial understanding capabilities, and trained on TSU tasks, enhances the generalized table reasoning capabilities of both models - Table-LLaVA and InternVL.

The effectiveness of the MCTR task is studied in Tables 19, 20 and 21. Inclusion of 50% masking between text and image modalities seems to be a hindrance for text-image alignment and hence affecting Table Structure Understanding and Table Question Answering capacity of the LLaVA model. It does lead some alignment improvements, reflected by marginal increase in metrics for some tests like TCL and RCE and Table Generation. Proper hyperparameter search for the masking ratio will lead to better text-image alignment and hence TSU and VQA capabilities of existing models.

VLM	MCTR	TSD	TCE	TCL	MCD	RCE	TSD	TCE	TCL	RCE
		Held-in Dataset					Held-out Dataset			
		Row/Col Acc	Acc	Acc	F1	Row/Col F1	Row/Col Acc	Acc	Acc	Row/Col F1
Table-LLaVA (baseline)	✗	0.460 0.755	0.309	0.452	0.411	0.368 0.500	0.241	0.364	0.223 0.267	0.368 0.500
Table-LLaVA (ours)	✓	0.457 0.750	0.305	**0.452**	0.407	0.356 **0.50**	0.232	**0.374**	**0.242** 0.259	0.356 **0.50**

Table 19 - Evaluation results on Table Structure Understanding tasks with models trained with and without the MCTR pretraining task.

VLM	MCTR	Table Reconstruction			Sub-Table retrieval	Row Insertion	Row Deletion	Row Update
		MMTab-Test			Tab-Bench			
		HTML (TEDS)	Markdown (TEDS)	Latex (TEDS)	HTML (TEDS)	HTML (TEDS)	HTML (TEDS)	HTML (TEDS)
Table-LLaVA (baseline)	✗	0.632	0.658	0.639	0.883	0.912	0.899	0.907
Table-LLaVA (ours)	✓	**0.635**	0.651	**0.646**	**0.884**	**0.917**	**0.901**	**0.913**

Table 20 - Evaluation results on Table Generation tasks with models trained with and without the MCTR Pretraining

VLM	MCTR	Question Answering					Fact Verification		Text Generation		
		Tab MWP (Acc)	WTQ (Acc)	HiTab (Acc)	TAT-QA (Acc)	FeTa QA (Bleu)	Tab Fact (Acc)	Info Tabs (Acc)	Hi Tab-T2T (Bleu)	Roto wire (Bleu)	Wiki Bio (Bleu)
Table-LLaVA (baseline)	✗	0.862	0.207	0.117	0.263	29.93	0.634	0.697	9.84	9.84	11.31
Table-LLaVA (ours)	✓	**0.871**	0.196	0.111	0.252	29.87	**0.635**	0.693	9.67	**11.14**	**11.48**

Table 21 - Evaluation results on academic Table VQA datasets with and without the MCTR Pretraining

3.7.8 - Conclusion

The PIN adapter, being a lightweight yet efficient solution, improves the localisation capabilities, and thus enhances generalised Table VQA over both LLaVA and Intern-VL baselines. InternVIT-300M is a much better vision encoder than CLIP-VIT-336/14, due to distillation from InternVIT-6B, resulting in better native OCR and TSU capabilities. PIN + InternVIT-300M leads to good quality vision embeddings which seems to be beneficial for the common vicuna-1.5 7B backbone for improving TSU and TQA. MCTR with 50% masking seems to be an obstacle while aligning the image and text modalities. It does lead some alignment improvements, reflected by marginal increase in metrics for some tests. A proper hyperparameter search is required to fetch the perfect masking ratio that will lead to better text-image alignment and improved TSU and Table VQA capabilities of existing baselines.

3.7.9 - Future Scope

The introduction of the PIN adapter helps to improve the Table VQA capabilities of existing VLMs. A proper investigation is required to overcome the obstacles faced during the MCTR pre-training.

Experimentation with larger 6B parameter VITs, and combination of PIN adapter + optimal MCTR should be explored. Patchwise PIN embeddings, for large VLMs with patchwise vision encoders like InternVL-2.0 should lead to further enhancement of table reasoning.

Another possible domain for exploration is Multimodal Table Understanding, where we provide the table in both the image and text modalities. The image modality can be used to interpret spatial information, hierarchical structure and other visual cues, while the text modality can be used as oracle/OCR data, and will remove the OCR overhead and vulnerability of VLMs while reading text data from the tables. This idea is visualized in Figure 22.

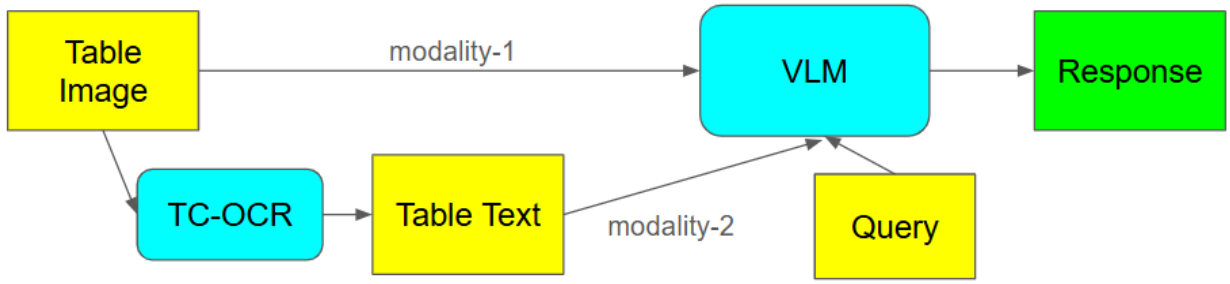


Fig 22 - Proposed pipeline for Multimodal Table Reasoning with VLMs

REFERENCES

[1] H. Liu, C. Li, Q. Wu, and Y. J. Lee, ‘Visual instruction tuning’, *Advances in neural information processing systems*, vol. 36, 2024.

[2] X. Zhong, J. Tang, and A. J. Yepes, ‘PubLayNet: largest dataset ever for document layout analysis’, in *2019 International Conference on Document Analysis and Recognition (ICDAR)*, 2019, pp. 1015–1022.

[3] A. Mondal, P. Lipps, and C. V. Jawahar, ‘IIIT-AR-13K: A new dataset for graphical object detection in documents’, in *Document Analysis Systems: 14th IAPR International Workshop, DAS 2020, Wuhan, China, July 26--29, 2020, Proceedings 14*, 2020, pp. 216–230.

[4] B. Pfitzmann, C. Auer, M. Dolfi, A. S. Nassar, and P. Staar, ‘Doclaynet: A large human-annotated dataset for document-layout segmentation’, in *Proceedings of the 28th ACM SIGKDD conference on knowledge discovery and data mining*, 2022, pp. 3743–3751.

[5] J. Redmon, ‘You only look once: Unified, real-time object detection’, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016.

[6] R. Girshick, J. Donahue, T. Darrell, and J. Malik, ‘Rich feature hierarchies for accurate object detection and semantic segmentation’, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580–587.

[7] S. Ren, K. He, R. Girshick, and J. Sun, ‘Faster R-CNN: Towards real-time object detection with region proposal networks’, *IEEE transactions on pattern analysis and machine intelligence*, vol. 39, no. 6, pp. 1137–1149, 2016.
[8] B. Smock and R. Pesala, *Table Transformer*. 06 2021.

[9]B. Smock, R. Pesala, and R. Abraham, ‘PubTables-1M: Towards comprehensive table extraction from unstructured documents’, in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2022, pp. 4634–4642.

[11] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, ‘End-to-end object detection with transformers’, in *European conference on computer vision*, 2020, pp. 213–229.

[12] D. Prasad, A. Gadpal, K. Kapadni, M. Visave, and K. Sultanpure, ‘CascadeTabNet: An approach for end to end table detection and structure recognition from image-based documents’, in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 572–573.

[13] Z. Cai and N. Vasconcelos, ‘Cascade r-cnn: Delving into high quality object detection’, in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 6154–6162.

[14] J. Wang *et al.*, ‘Deep high-resolution representation learning for visual recognition’, *IEEE transactions on pattern analysis and machine intelligence*, vol. 43, no. 10, pp. 3349–3364, 2020.

[15] Y. N. Du *et al.*, ‘PP-OCRv2: Bag of tricks for ultra lightweight OCR system. arXiv 2021’, *arXiv preprint arXiv:2109. 03144*.

[16] M. Li, L. Cui, S. Huang, F. Wei, M. Zhou, and Z. Li, ‘TableBank: A Benchmark Dataset for Table Detection and Recognition’, *arXiv [cs.CV]*. 2019.

[17] W. Chen *et al.*, ‘Tabfact: A large-scale dataset for table-based fact verification’, *arXiv preprint arXiv:1909. 02164*, 2019.

[18] P. Pasupat and P. Liang, ‘Compositional semantic parsing on semi-structured tables’, *arXiv preprint arXiv:1508. 00305*, 2015.
[19] L. Nan *et al.*, ‘FeTaQA: Free-form table question answering’, *Transactions of the Association for Computational Linguistics*, vol. 10, pp. 35–49, 2022.

[20] T. Zhang, X. Yue, Y. Li, and H. Sun, ‘Tablellama: Towards open large generalist models for tables’, *arXiv preprint arXiv:2311. 09206*, 2023.

[21] P. P. Liang *et al.*, ‘Multiviz: Towards visualizing and understanding multimodal models’, *arXiv preprint arXiv:2207. 00056*, 2022.

[22] Y. Sui, M. Zhou, M. Zhou, S. Han, and D. Zhang, ‘Table meets llm: Can large language models understand structured table data? a benchmark and empirical study’, in *Proceedings of the 17th ACM International Conference on Web Search and Data Mining*, 2024, pp. 645–654.

[23] A. P. Parikh *et al.*, ‘ToTTo: A controlled table-to-text generation dataset’, *arXiv preprint arXiv:2004. 14373*, 2020.

[24] W. Chen, H. Zha, Z. Chen, W. Xiong, H. Wang, and W. Wang, ‘Hybridqa: A dataset of multi-hop question answering over tabular and textual data’, *arXiv preprint arXiv:2004. 07347*, 2020.

[25] M. Iyyer, W.-T. Yih, and M.-W. Chang, ‘Search-based neural structured learning for sequential question answering’, in *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 2017, pp. 1821–1831.

[26] R. Aly *et al.*, ‘Feverous: Fact extraction and verification over unstructured and structured information’, *arXiv preprint arXiv:2106. 05707*, 2021.

[27] A. Dubey *et al.*, ‘The llama 3 herd of models’, *arXiv preprint arXiv:2407. 21783*, 2024.

[28] G. Team *et al.*, ‘Gemini: a family of highly capable multimodal models’, *arXiv preprint arXiv:2312. 11805*, 2023

[29] J. Achiam *et al.*, ‘Gpt-4 technical report’, *arXiv preprint arXiv:2303. 08774*, 2023.

[30] M. Dorkenwald, N. Barazani, C. G. M. Snoek, and Y. M. Asano, ‘PIN: Positional Insert Unlocks Object Localisation Abilities in VLMs’, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 13548–13558.

[31] J. Li, D. Li, S. Savarese, and S. Hoi, ‘Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models’, in International conference on machine learning, 2023, pp. 19730–19742.

[32] A. Awadalla et al., Open-flamingo: An open-source framework for training large autoregressive vision-language models’, arXiv preprint arXiv:2308. 01390, 2023.

[33] J. Xiao, A. Yao, Y. Li, and T.-S. Chua, ‘Can I trust your answer? visually grounded video question answering’, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 13204–13214.

[34] Z. Chen et al., ‘Internvl: Scaling up vision foundation models and aligning for generic visual-linguistic tasks’, in Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition, 2024, pp. 24185–24198.

[35] Z. Chen et al., ‘How far are we to gpt-4v? closing the gap to commercial multimodal models with open-source suites’, arXiv preprint arXiv:2404. 16821, 2024.

[36] Vicuna. Vicuna: An open-source chatbot impressing gpt-4 with 90%* chatgpt quality. https://vicuna.lmsys.org/, 2023.

APPENDIX

A1 -

1) Row and Column Size

```
<Instruction> This is a table structure understanding task. The table rows and columns are indexed from 0. The first row contains the column headers. Read the task description and answer the question.
<Task Name> table_size
<Task Definition> Find the number of rows and columns present in the table
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points | pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 | 13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question> How many rows and columns are present in the table? Return the answer as (number_of_rows, number_of_columns). Only return the answer without any other information.
 Return only the answer and nothing else
<Answer>
```

2) Cell Lookup

```
<Instruction> This is a table structure understanding task. The table rows and columns are indexed from 0. The first row contains the column headers. Read the task description and answer the question.
<Task Name> cell_lookup
<Task Definition> Find the row_index and column_index of the table cell containing the input text
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points | pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 | 13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
```

```
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question> What is the position of the table cell containing '10th'. Return the
answer in the format (row_index,column_index). If there are multiple answers,
output them separated by a '|'. Only output the answer without any other
information.
 Return only the answer and nothing else
<Answer>['6', '8']
<Question> What is the position of the table cell containing '10th'. Return the
answer in the format (row_index,column_index). If there are multiple answers,
output them separated by a '|'. Only output the answer without any other
information.
 Return only the answer and nothing else
<Answer>
```

3) Reverse Cell Lookup

```
<Instruction> This is a table structure understanding task. The table rows and
columns are indexed from 0. The first row contains the column headers. Read the
task description and answer the question.
<Task Name> reverse_cell_lookup
<Task Definition> Find the cell value at a given the row_index and column_index
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points
| pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 |
13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question>  What is the cell value at the given row_index '6' and column_index '8'.
Only output the cell value without any other information
 Return only the answer and nothing else
<Answer>10th
<Question>  What is the cell value at the given row_index '6' and column_index '8'.
Only output the cell value without any other information
 Return only the answer and nothing else
<Answer>
```

4) Column Retrieval

```
<Instruction> This is a table structure understanding task. The table rows and
```

```
columns are indexed from 0. The first row contains the column headers. Read the
task description and answer the question.
<Task Name> column_retrieval
<Task Definition> Find the column name at a given column_index
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points
| pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 |
13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question> What is the column name with  column_index '7' of the following table?
Only give the column name without any explanation
 Return only the answer and nothing else
<Answer>points
<Question> What is the column name with  column_index '0' of the following table?
Only give the column name without any explanation
 Return only the answer and nothing else
<Answer>
```

5) Row Retrieval

```
<Instruction> This is a table structure understanding task. The table rows and
columns are indexed from 0. The first row contains the column headers. Read the
task description and answer the question.
<Task Name> row_retrieval
<Task Definition> Find the cell values at a given row_index
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points
| pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 |
13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question> What are the cell values of the row with row_index '2' ? Only list the
cell values one by one using '|' to split the cell values. Only return the answer
without any other information.
```

```
 Return only the answer and nothing else
<Answer>['2008 - 09', 'a1 grand prix', 'a1 team mexico', '4', '0', '0', '0', '19',
'13th (1)']
<Question> What are the cell values of the row with row_index '5' ? Only list the
cell values one by one using '|' to split the cell values. Only return the answer
without any other information.
 Return only the answer and nothing else
<Answer>
```

6) Sub Table Retrieval

```
<Instruction> This is a table structure understanding task. The table rows and
columns are indexed from 0. The first row contains the column headers. Read the
task description and answer the question.
<Task Name> sub_table_retrieval
<Task Definition> Fetch the portion of the table present within the given row and
column indices
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points
| pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 |
13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question> Return a new table containing the portion of the input table enclosed
within the row_indexes '1' and '6' and column_indexes '3' and '8'. Also include the
corresponding column header from the input table as the first row of the new table.
Return the new table in the csv format. Only return the answer without any other
information.
 Return only the answer and nothing else
<Answer>races,wins,poles,fast laps,points,pos
2,0,0,0,8,17th
4,0,0,0,19,13th (1)
6,0,0,0,94,13th
10,0,0,0,5,16th (1)
12,0,0,0,78,17th
14,0,2,0,42,10th

<Question> Return a new table containing the portion of the input table enclosed
within the row_indexes '3' and '5' and column_indexes '1' and '2'. Also include the
corresponding column header from the input table as the first row of the new table.
Return the new table in the csv format. Only return the answer without any other
```

```
information.
 Return only the answer and nothing else
<Answer>
```

7) Column Range Retrieval

```
<Instruction> This is a table structure understanding task. The table rows and
columns are indexed from 0. The first row contains the column headers. Read the
task description and answer the question.
<Task Name> column_range_retrieval
<Task Definition> Find the range of values present in a particular column
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points
| pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 |
13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question> Return the minimum and maximum value present in the column with
column_name 'fast laps'. Return the answer as (min_value, max_value). Only return
the answer without any other information.
 Return only the answer and nothing else
<Answer>['0', '0']
<Question> Return the minimum and maximum value present in the column with
column_name 'fast laps'. Return the answer as (min_value, max_value). Only return
the answer without any other information.
 Return only the answer and nothing else
<Answer>
```

8) Row Range Retrieval

```
<Instruction> This is a table structure understanding task. The table rows and
columns are indexed from 0. The first row contains the column headers. Read the
task description and answer the question.
<Task Name> row_range_retrieval
<Task Definition> Find the range of values present in a particular row
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points
| pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
```

```
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 |
13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question> Return the minimum and maximum value present in the row with row_index
'4'. Consider only integer and fractional data. Return the answer as (min_value,
max_value). Only return the answer without any other information.
 Return only the answer and nothing else
<Answer>['0', '10']
<Question> Return the minimum and maximum value present in the row with row_index
'5'. Consider only integer and fractional data. Return the answer as (min_value,
max_value). Only return the answer without any other information.
 Return only the answer and nothing else
<Answer>
```

9) Row Insertion

```
<Instruction> This is a table structure understanding task. The table rows and
columns are indexed from 0. The first row contains the column headers. Read the
task description and answer the question.
<Task Name> insert_row
<Task Definition> Insert a new row at a particular row_index
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points
| pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 |
13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Input Row> ['2008', 'champ car atlantic', 'eurointernational', 6, 0, 0, 0, 19,
'17th']
<Question> Insert the given input row into the input table after the row with
row_index '5'. Return the new table in the csv format. Only return the answer
without any other information.
 Return only the answer and nothing else
<Answer>season,series,team,races,wins,poles,fast laps,points,pos
2009,champ car atlantic,jensen motorsport,2,0,0,0,8,17th
2008 - 09,a1 grand prix,a1 team mexico,4,0,0,0,19,13th (1)
2008,champ car atlantic,forsythe championship racing,6,0,0,0,94,13th
```

```
2007 - 08,a1 grand prix,a1 team mexico,10,0,0,0,5,16th (1)
2007,champ car atlantic,us racetronics,12,0,0,0,78,17th
2008,champ car atlantic,eurointernational,6,0,0,0,19,17th
2006,formula bmw usa,eurointernational,14,0,2,0,42,10th

<Question> Insert the given input row into the input table after the row with
row_index '2'. Return the new table in the csv format. Only return the answer
without any other information.
 Return only the answer and nothing else
<Answer>
```

10) Row Deletion

```
<Instruction> This is a table structure understanding task. The table rows and
columns are indexed from 0. The first row contains the column headers. Read the
task description and answer the question.
<Task Name> delete_row
<Task Definition> Delete a row at a particular row_index
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points
| pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 |
13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Question> Delete the row with row_index '6'. Return the new table in the csv
format. Only return the answer without any other information.
 Return only the answer and nothing else
<Answer>season,series,team,races,wins,poles,fast laps,points,pos
2009,champ car atlantic,jensen motorsport,2,0,0,0,8,17th
2008 - 09,a1 grand prix,a1 team mexico,4,0,0,0,19,13th (1)
2008,champ car atlantic,forsythe championship racing,6,0,0,0,94,13th
2007 - 08,a1 grand prix,a1 team mexico,10,0,0,0,5,16th (1)
2007,champ car atlantic,us racetronics,12,0,0,0,78,17th

<Question> Delete the row with row_index '1'. Return the new table in the csv
format. Only return the answer without any other information.
 Return only the answer and nothing else
<Answer>
```

11) Row Update

<Instruction> This is a table structure understanding task. The table rows and columns are indexed from 0. The first row contains the column headers. Read the task description and answer the question.
<Task Name> update_row
<Task Definition> Update the cell values at a particular row_index
<Input Table Format> markdown
<Input Table> | season | series | team | races | wins | poles | fast laps | points | pos |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 2009 | champ car atlantic | jensen motorsport | 2 | 0 | 0 | 0 | 8 | 17th |
| 2008 - 09 | a1 grand prix | a1 team mexico | 4 | 0 | 0 | 0 | 19 | 13th (1) |
| 2008 | champ car atlantic | forsythe championship racing | 6 | 0 | 0 | 0 | 94 | 13th |
| 2007 - 08 | a1 grand prix | a1 team mexico | 10 | 0 | 0 | 0 | 5 | 16th (1) |
| 2007 | champ car atlantic | us racetronics | 12 | 0 | 0 | 0 | 78 | 17th |
| 2006 | formula bmw usa | eurointernational | 14 | 0 | 2 | 0 | 42 | 10th |

<Input Cells> {'wins': '0', 'pos': '13th', 'races': '14'}
<Question> Update the specified input cells at the row with row_index '1'. Return the new table in the csv format. Only return the answer without any other information.
 Return only the answer and nothing else
<Answer>season,series,team,races,wins,poles,fast laps,points,pos
2009,champ car atlantic,jensen motorsport,14,0,0,0,8,13th
2008 - 09,a1 grand prix,a1 team mexico,4,0,0,0,19,13th (1)
2008,champ car atlantic,forsythe championship racing,6,0,0,0,94,13th
2007 - 08,a1 grand prix,a1 team mexico,10,0,0,0,5,16th (1)
2007,champ car atlantic,us racetronics,12,0,0,0,78,17th
2006,formula bmw usa,eurointernational,14,0,2,0,42,10th

<Question> Update the specified input cells at the row with row_index '1'. Return the new table in the csv format. Only return the answer without any other information.
 Return only the answer and nothing else
<Answer>