UniTable: Towards a Unified Framework for Table Recognition via Self-Supervised Pretraining

ShengYun Peng[1] **Aishwarya Chakravarthy**[1] **Seongmin Lee**[1] **Xiaojing Wang**[2]
Rajarajeswari Balasubramaniyan[2] **Duen Horng Chau**[1]
[1]Georgia Tech [2]ADP, Inc.
{speng65,achakrav6,seongmin}@gatech.edu
{xiaojing.wang,raji.balasubramaniyan}@adp.com

Abstract

Tables convey factual and quantitative data with implicit conventions created by humans that are often challenging for machines to parse. Prior work on table recognition (TR) has mainly centered around complex task-specific combinations of available inputs and tools. We present **UniTable**, a training framework that unifies both the **training paradigm** and **training objective** of TR. Its training paradigm combines the simplicity of purely pixel-level inputs with the effectiveness and scalability empowered by self-supervised pretraining (SSP) from diverse unannotated tabular images. Our framework unifies the training objectives of all three TR tasks — extracting table structure, cell content, and cell bounding box (bbox) — into a unified task-agnostic training objective: language modeling. Extensive quantitative and qualitative analyses highlight UniTable's state-of-the-art (SOTA) performance on four of the largest TR datasets. UniTable's table parsing capability has surpassed both existing TR methods and general large vision-language modelss (VLMs), *e.g.*, GPT-4o, GPT-4-turbo with vision, and LLaVA. Our code is publicly available at `https://github.com/poloclub/unitable`, featuring a Jupyter Notebook that includes the complete inference pipeline, fine-tuned across multiple TR datasets, supporting all three TR tasks.

1 Introduction

Tables are ubiquitous in documents, as they serve to summarize factual and quantitative data — information that is cumbersome to describe in text but nevertheless crucial [10, 8]. Due to the implicit conventions used by humans in creating tables, the representations within tables are often challenging for machines to parse. Even the milestone VLM, GPT-4o [29], GPT-4V(vision) [39] and LLaVA [21], still struggles with various document-related tasks including TR. GPT-4V tends to omit content in large tables and performs worse when faced with complex tables, *e.g.*, spanning or empty cells and uneven text distributions [35].

Prior work on TR has mainly centered around complex task-specific combinations of available inputs and tools. Typically, table structure was predicted by an image-to-text pipeline [42] and cell bbox was predicted by a detection head, *e.g.*, Faster R-CNN [36] or DETR [28]. The assumption regarding predicting cell content varies: some studies assume the presence of a portable document format (PDF) accompanying the tabular image [28], while others rely on external text line detection and text recognition models [40, 12]. However, training generic Transformers under a single language modeling objective, *i.e.*, predicting the next token, for diverse tasks has achieved remarkable success across diverse tasks in language [32, 3], vision [4, 5], and vision-and-language domains [1]. We wonder whether understanding visual language in tables, *i.e.*, extracting table structure, cell content, and cell bbox from tabular images, can be seamlessly integrated into the language modeling training

Preprint. Under review.

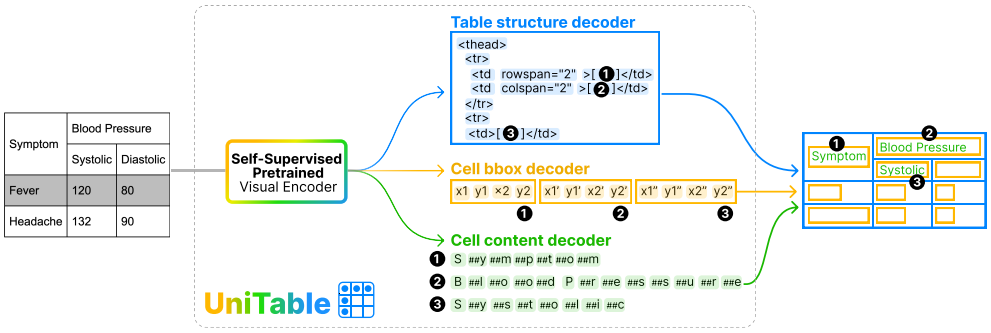


Figure 1: **UniTable**, a training framework that unifies both **training paradigm** and **training objective** of TR. In UniTable, the visual encoder is self-supervised pretrained and then finetuned along with the task decoder on supervised datasets. UniTable unifies the training objectives of all three TR tasks — extracting table structure, cell bbox, and cell content — into a unified task-agnostic training objective: language modeling. With UniTable, the user inputs a tabular image and obtains the corresponding digitalized table in HTML.

framework. This integration is challenging because: (1) the fusion of vision and language in tabular images demands high-fidelity reading and rich high-level representations [15]; (2) the large amount of unannotated tabular images in practice cannot be leveraged by existing supervised learning approaches [28, 26]; (3) the diverse output of table-related tasks are typically addressed by task-specific models [12]; and (4) direct application of a linear projection Transformer results in the significant performance drop [30], leading prior work to exclusively employ a hybrid convolutional neural network (CNN)-Transformer architecture. We resolve the above challenges by proposing **UniTable**, a training framework that unifies both **training paradigm** and **training objective** of TR and make the following major contributions (Fig. 1):

1. **UniTable's training paradigm combines the simplicity of purely pixel-level inputs with the effectiveness and scalability empowered by SSP from diverse unannotated tabular images.** Specifically, UniTable unifies the training paradigm for TR: pretraining the visual encoder by predicting the masked tabular images in a self-supervised manner and finetuning the visual encoder along with the task decoder on supervised datasets. With UniTable, the table structure prediction on SynthTabNet [28], a comprehensive dataset with 600k tables across finance, marketing, and academia in both dense and sparse format, achieves the SOTA 99.18% when self-supervised pretrained on 2M images, significantly lifting the original accuracy of 84.04% when trained from scratch using a linear projection Transformer. Owing to the powerful SSP, UniTable has also successfully mitigated the performance drop caused by replacing the CNN backbone with the linear projection.
2. **UniTable unifies the training objectives of all three TR tasks — extracting table structure, cell content, and cell bbox — into a unified task-agnostic training objective: language modeling.** Specifically, the input to our model is an image in the form of raw pixels only and the output is text in the form of token sequences, and the training objective is language modeling. UniTable's

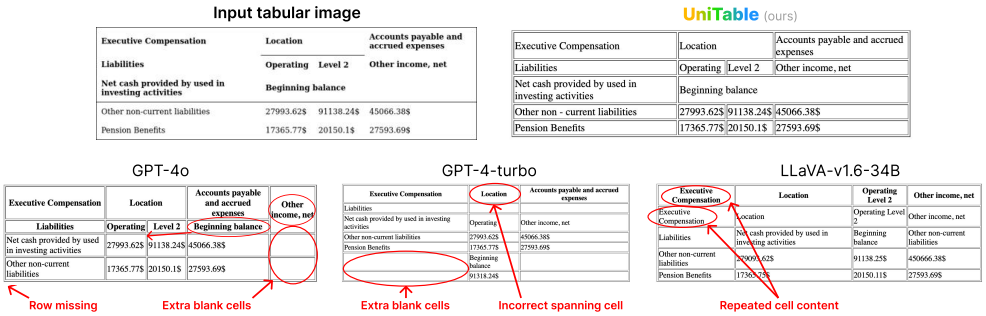


Figure 2: The table parsing capability of UniTable has surpassed that of general large VLMs, *e.g.*, GPT-4o, GPT-4-turbo with vision, and LLaVA. For complex tables that include multiple spanning cells, UniTable can successfully reconstruct the table in HTML, whereas general large VLMs fail in various aspects.

SOTA performance on the FinTabNet dataset [41] demonstrates our approach's generalizability to the PDF input modality as we can simply convert PDF to images. Our framework also enables us to leverage the power of SSP on large-scale unannotated tabular images as all models are finetuned from SSP. UniTable's unified training objective applies to both linear projection Transformer and hybrid CNN-Transformer architectures conventionally used in TSR.

3. **Extensive quantitative and qualitative analyses highlight UniTable's SOTA performance on four of the largest TR datasets**: ICDAR 2019 B2 Modern [8], PubTabNet [42], FinTabNet [41], and SynthTabNet [28]. UniTable's table parsing capability has surpassed both existing TR methods and general large VLMs (Fig. 2), *e.g.*, GPT-4o, GPT-4V(ision), and LLaVA. Due to our unified language modeling framework formulation, we discover three types of previously unacknowledged inconsistencies in the groundtruth annotations of PubTables-1M [36], one of the largest TSR datasets, accounting for more than 53.10% of its training set. Our visualization of the visual tokens reveals the key reason for why SSP works — the visual semantics captured by the visual codebook show a fine-grained categorization to represent the implicit human conventions used when creating the tables.
4. **Open-source code and UniTable in practice.** To promote reproducible research, enhance transparency, SOTA innovations, and facilitate fair comparisons in our domain as tables are a promising modality for representation learning, we open-source our code (anonymized)[1]. We provide all the details regarding training, validation, testing, and ablation studies. To enable users to easily try UniTable on their own tabular images and obtain fully digitized HTML tables, we release the first-of-its-kind Jupyter Notebook of the whole inference pipeline, fine-tuned across multiple TR datasets, supporting all three TR tasks.

2 Background

2.1 Task Definition

The goal of TR is to translate the input tabular image $\mathbf{I}$ into a machine-readable sequence $\mathbf{T}$, which typically consists of table structure $\mathbf{S}$, table cell bbox $\mathbf{B}$, and table cell content $\mathbf{C}$. Structure $\mathbf{S} = [s_1, ..., s_m]$ is a sequence of tokenized HTML table tags s, cell bbox $\mathbf{B} = [\mathbf{b}_1, ..., \mathbf{b}_n]$ is a sequence of bboxes defined by $\mathbf{b} = (x_{min}, y_{min}, x_{max}, y_{max})$, and cell content $\mathbf{C} = [c_1, ..., c_n]$ comprises the content within each cell in reading order. Note that the sequence length of $\mathbf{B}$ and $\mathbf{C}$ are the same, but shorter than $\mathbf{S}$ as the HTML tags contain both empty and non-empty cells. Since each cell i is defined by a single bbox, c_i can have either single or multiple text lines.

2.2 Model Architecture

TR model architecture has two modules: visual encoder and task decoder. The visual encoder extracts features from input $\mathbf{I}$, and the task decoder predicts $\mathbf{T}$. For the visual encoder, prior work employed either an off-the-shelf CNN backbone, *e.g.*, ResNets and ResNet variants [11], or hybrid CNN-Transformer architecture. EDD [42] explored five different ResNet-18 variants, TableMaster [24] combined residual blocks with multi-aspect global context attention, TableFormer [28] connected a ResNet-18 with Transformer encoder layers, and VAST [12] adopted the first four stages of a ResNet-31. These convolution layers cannot be replaced because a direct employment of a vanilla Transformer with linear projection leads to a significant performance drop [30]. However, linear projection, that divides image into patches, is a widely used input image processor in SOTA vision Transformers [7, 22], VLMs [16, 20] and multi-modal models [27]. Recent work on visual language understanding has also successfully adopted the linear projection [14, 15]. Thus, to avoid having a separate architecture design solely for the table domain, we aim to keep the linear projection Transformer, and mitigate the performance gap by SSP. Sec. 3 shows that pretraining the visual encoder in a self-supervised manner significantly helps the model learn how to parse the table structure and achieves performance even higher than architectures with convolutions.

[1] https://github.com/poloclub/unitable

3 A Unified Framework for Pretraining and Finetuning TR Models

We pretrain the visual encoder by predicting the masked tabular images in a self-supervised manner and finetune the visual encoder along with the task decoder using the supervised dataset for each task, as shown in Fig. 1. Sec. 3.1 introduces the SSP of the visual encoder, and Sec. 3.2 details how we unify all TR tasks and finetune with the pretrained visual encoder.

3.1 Self-Supervised Pretraining of the Visual Encoder

Before pretraining, each tabular image I is tokenized into discrete visual tokens. During pretraining, I is divided into patches, and a portion of the image patches are masked so that the visual encoder predicts which visual token is chosen to replace the masked regions.

Image tokens. Define a visual codebook in the latent space $Z \in \mathbb{R}^{K \times D}$, representing K entries of visual tokens $z_i \in \mathbb{R}^D$. The visual codebook is trained in such a way that an input image, once it is embedded into an image grid, each pixel on the embedded image grid can be substituted with a visual token from the codebook. Decoding this modified image grid will then reconstruct the input image. We use Vector Quantized-Variational AutoEncoder (VQ-VAE) [37] to train the visual codebook. Specifically, the tabular image I is tokenized into discrete tokens z after passing through the encoder $q_\phi(z|I)$, and the decoder $p_\psi(I|z)$ takes these discrete tokens and rebuilds the original image. The training objective of VQ-VAE is to maximize the $\mathbb{E}_{z \sim q_\phi(z|I)}[\log p_\psi(I|z)]$ with respect to ϕ and ψ. The training is non-differentialble due to the categorical distribution in selecting visual tokens. Thus, we use Gumbel-Softmax [13] as a reparameterization trick following DALL-E [33]. We have trained the VQ-VAE on 1M and 2M tabular images, where $K = 8192$ for 1M, and $K = 16384$ for 2M.

Image patches. Given an input tabular image $I \in \mathbb{R}^{H \times W \times C}$, the linear projection divides I into a sequence of flattened 2D patches $I_p \in \mathbb{R}^{N \times (P^2 \cdot C)}$, where C is the number of channels, (P, P) is the size of each image patch, and $N = HW/P^2$ is the number of patches. It is implemented by a kernel $P \times P$, stride P convolution. We set $P = 16$ and $I \in \mathbb{R}^{448 \times 448 \times 3}$, thus the sequence length of the image patches is 28×28. Approximately 40% of the sequence is replaced with a masked token and the pretraining objective is to maximize the log-likelihood of the visual tokens of the masked region given the unmasked region. The image tokenizer's codebook provides the groundtruth visual tokens.

The pretraining task is inspired by the success of masked language [6] and natural image [2] modeling, but our work differs in the following ways:

(1) Table incorporates both vision and language representation of data, presenting concise human language with implicit conventions. Models for visual language understanding are required to read with high fidelity while also building rich high-level representation, relying on signals from both vision and language. In contrast, our work first explores the feasibility of self-supervised learning only on images for visual language tasks, using tables as an example.

(2) A domain shift from semantic-rich natural images to predominantly black texts on white background tabular images poses a difficult optimization problem. It appears that tabular images are mainly text and lines separating rows and columns, so the visual codebook can quickly exhaust the table patterns with minimal tokens. Contrarily, the training of VQ-VAE can easily diverge. We discover that training stability is achieved by increasing the total number of tokens or introducing tabular images with colorful backgrounds. Sec. 5.3 visualizes the semantics of the trained visual codebook and provides an explanation for the substantial number of tokens required.

3.2 Unified Finetuning Framework

Prior work employed a task-specific decoder for each task, where $\mathbf{S}$ was predicted by an image-to-text pipeline and $\mathbf{B}$ was predicted by a detection head, *e.g.*, Faster R-CNN [36] or DETR [28]. The assumption on predicting $\mathbf{C}$ varies: some assume a PDF is always accompanying the tabular image [28], others rely on external text line detection and text recognition models [40, 12]. We aim to provide a unified task-agnostic training framework, where the input to our model is an image in the form of raw pixels only and the output is text in the form of token sequences. This setting is also generalizable to PDF input modality as we can simply take a screenshot of the PDF. The framework also enables us to leverage the visual encoder pretrained on the unannotated tabular images.

Table 1: UniTable outperforms prior methods and achieves SOTA on four out of five largest publicly available TR datasets across all available tasks. Our method is trained with a task-agnostic language modeling loss and does not rely on external PDF for text extraction and bbox post-processing.

	IC19B2M		PubTabNet			FinTabNet	SynthTabNet		PubTables-1M	
	IoU 0.6	WAvg. F1	AP_{50}	S-TEDS	TEDS	S-TEDS	AP_{50}	S-TEDS	AP_{50}	AP_{75}
SOTA	GTE	GTE	VAST	VAST	VAST	VAST	TableFormer	DRCC	DETR	DETR
	38.50	24.80	94.50	97.23	96.31	98.63	87.70	98.70	**97.10**	**94.80**
UniTable										
Base	54.97	40.15	97.94	95.63	94.78	97.19	98.99	98.97	94.48	88.64
Large	**58.10**	**42.62**	**98.43**	**97.89**	**96.50**	**98.89**	**99.00**	**99.39**	95.68	93.28

Table structure $\mathbf{S}$. Predicting the table structure $\mathbf{S}$ already fits our training framework as $\mathbf{S}$ is defined by discrete HTML table tags. For non-spanning cells, we use `<td></td>` and `<td>[]</td>` to denote empty and non-empty cells. For spanning cells, `<td` marks the beginning, and `></td>` and `>[]</td>` marks the ending of empty and non-empty cells. The specific tokens for spanning cells are `rowspan="n"` and `colspan="n"`. We use $n \in [2, 19]$ as that covers most of the tables in practice. Apart from the data cell tags, the vocabulary also contains the following tags that define a table: `<thead>`, `<tbody>`, `<tr>`, and their corresponding closing tags.

Table cell bbox $\mathbf{B}$. Each cell bbox $\mathbf{b}$ are four continuous coordinates, which are not naturally expressed as discrete tokens. Inspired by Pix2Seq [4], we discretize the coordinate into an integer between 0 and image size. The two directions of an image share the same vocabulary. Since we need to predict all bboxes within a tabular image, each quantized bbox is concatenated together in reading order: from left to right and top to bottom. This formulation completes the quantization and serialization of all bboxes into a sequence of discrete tokens. At inference time, we de-serialize the predicted sequence into groups of four tokens.

Table cell content $\mathbf{C}$. After predicting all bbox coordinates, we only need to perform optical character recognition (OCR) on the image region within each bbox. Note that each cell is defined by a single bbox, thus the cell content can have single or multiple lines of text. In the training stage, the model is trained on a mixture of single line and multi-line dataset. At inference time, we parse all text simultaneously as each cell bbox is independent. Finally, we insert the cell content back into the non-empty cells `<td>[]</td>` or `>[]</td>` as the reading order is already preserved in both $\mathbf{S}$ and $\mathbf{B}$. We use WordPiece tokenizer [38] with charater-level granularity since OCR requires the model to read instead of understanding the semantics, which significantly reduces the total vocabulary size to less than 6k.

Up till now, we have completely digitalized a tabular image into HTML with a unified image-to-text framework. Note that all visual encoders are initialized from SSP. For cell content and cell bbox, there is an alternative solution that we first generate all the cell content within a table, and then predict the cell bbox via prompting the model with cell content. However, we find it hard for the model to predict all the cell content first as all tabular images are rescaled to a fixed size during augmentation, and such rescaling leads to texts in various aspect ratios. Thus, we do not use this solution and instead generate all cell bbox first.

Training Objective. Since all task outputs have been formulated into a sequence of discrete tokens, the objective function is simply the maximum likelihood of tokens conditioned on pixel inputs and the preceding tokens. Denote the probability of the ith step prediction $p(t_i|I, t_{1:i-1}; \theta)$, we directly maximize the correct structure prediction by using the following formulation:

$$\theta^* = \arg\max_{\theta} \sum_{(I,T)} \log p(T|I; \theta), \tag{1}$$

where θ are model parameters.

4 Experiments

4.1 Implementation

Architecture. We have trained two model variants: (1) a *base* model with 30M parameters including 4 encoder layers, 8 attention heads with a hidden size of 512, (2) a *large* model with 125M parameters including 12 encoder layers, 12 attention heads with a hidden size of 768. Both base and large models have a task decoder of 4 decoder layers. The maximum token sequence length is 512 for table structure, 1024 for cell bbox, and 200 for cell content, as we find such settings satisfy most tables.

Training and inference We have pretrained the VQ-VAE on 1M and 2M tabular images. The 1M VQ-VAE is trained on PubTabNet [42] and SynthTabNet [28], and the extra 1M datasets for training 2M VQ-VAE are PubTables-1M [36] and TableBank [18]. In Sec. 4.3, we present the finetuning results of 2M VQ-VAE for comparing with SOTA methods. In Sec. 5.1, we ablate the effectiveness and scalability of SSP on both 1M and 2M VQ-VAE. All models are trained with the AdamW optimizer [23]. We employ a cosine learning rate scheduler with a linear warmup. All models are trained for 24 epochs for a fair comparison. We apply teacher forcing during training and employ greedy decoding at inference time.

4.2 Evaluation Metrics

Cell adjacency relations (CAR) was first proposed in ICDAR2013 competition [10] and improved by ICDAR2019 competition [8]. It aligns the the predicted bbox with the groundtruth bbox for each table cell based on intersection over union (IoU) and generates a list of adjacency relations between a non-empty cell and its nearest horizontal and vertical neighbors. The precision, recall, and F1 score are computed based on this converted 1-D adjacency relation list.

COCO average prevision (AP) [19] is a widely used metric for generic object detection, which has been reported in other work for evaluating table cell detection. We use the COCO evaluation toolkit[2] and report mean AP (mAP), AP_{50}, and AP_{75}.

Tree-edit-distance-based similarity (TEDS) was created by PubTabNet [42] to robustify the CAR metric against cell shift perturbation and cell content perturbation. TEDS converts the table HTML code into a tree structure and measures the edit distance between the prediction T_{pred} and the groundtruth T_{gt}. A shorter edit distance indicates a higher degree of similarity, leading to a higher TEDS score. TEDS measures both the table structure and table cell content, and we use S-TEDS [12] when only the table structure is considered.

4.3 Results on Datasets

We evaluate UniTable on five of the largest publicly available TR datasets as shown in Table 1. Comparing UniTable with the prior SOTA on each dataset across all available tasks, we achieve new SOTA on four out of the five datasets even without training with task-specific loss [12] or relying on external PDF for text extraction and bbox post-processing [28]. Below is an introduction of each dataset and comparisons with previous methods.

ICDAR 2019 B2 Modern (IC19B2M) [8] was originated from ICDAR 2019 table competition. The dataset has two subsets for TR, archival and modern, and only the modern subset has table cell content bbox annotations. The competition computes CAR F1 score at IoU $\in [0.6, 0.7, 0.8, 0.9]$ and ranks method by weighted average F1 (WAvg. F1):

$$\text{WAvg. F1} = \frac{\sum_{i=1}^{4} \text{IoU}_i \times \text{F1@IoU}_i}{\sum_{i=1}^{4} \text{IoU}_i} \tag{2}$$

We evalute on all 100 test tables from the modern subset and report both WAvg. F1 and IoU 0.6 as in other work. Our method significantly improves the previous SOTA GTE [41] by a large margin.

PubTabNet [42] contains 509k images of heterogeneous tables extracted from the medical scientific articles. It is the first large-scale TR dataset that provides annotations (in HTML format) of table cell bbox, table structure, and table cell content. PubTabNet measures the table cell bbox by COCO AP_{50},

[2] https://github.com/cocodataset/cocoapi

table structure by S-TEDS, and full table including both structure and cell content by TEDS. The authors of PubTabNet also developed the EDD model, which consisted of a CNN encoder and dual recurrent neural network (RNN) decoders for predicting table structure and cell content, respectively. TableFormer [28] improved EDD by replacing the cell content decoder with a cell bbox decoder and extracted all contents from the PDF corresponding to the tabular image. VAST [12] added an auxiliary visual-alignment loss while training the cell bbox decoder and achieved previous SOTA on all three metrics. Our unified training framework with SSP achieves the new SOTA on all tasks even without leveraging any external PDF as PDF corresponding to the tabular image may not always exist. Specifically, both UniTable-base and UniTable-large outperform VAST on AP_{50} by more than 3 percentage points (pp), which confirms the effectiveness of converting the cell bbox detection to language modeling.

FinTabNet [41] is a dataset containing 113k tables from the annual reports of the S&P 500 companies in PDF format. The major challenge of this dataset is that financial tables largely differ from scientific and government document tables in that the former has fewer graphical lines, larger gaps within each table, and more color variations. Thus, FinTabNet mainly evaluates table structure prediction accuracy, *i.e.*, S-TEDS. VAST trained the model with an auxiliary supervised signal and achieved the previous SOTA. We achieve the new SOTA by leveraging SSP and finetuning without a task-specific loss objective. The performance gain from base to large shows that our method scales with model parameters.

SynthTabNet [28] is a large-scale synthetically generated dataset that offers control over 1) dataset size, 2) table structure, 3) table style, and 4) content type. The dataset aims to overcome the limitations of PubTabNet and FinTabNet, which are skewed table distributions towards simpler tables, limited variance in appearance styles, and restricted cell content domains. Thus, SynthTabNet is organized into 4 subsets of 150k tables (600k in total), namely Finance, PubTabNet, Marketing, and Sparse. The first two mimic the appearance of FinTabNet and PubTabNet but encompass more complex table structures. Marketing adopts a colorful appearance with high contrast that resembles real-world marketing tables, and Sparse contains tables with sparse content. The authors of SynthTabNet propose TableFormer as a baseline. TableFormer predicts table structure and cell bbox and relies on external PDF to extract cell content, so AP_{50} and S-TEDS are the evaluation metrics. Both our base and large models outperform previous SOTA on AP_{50} and S-TEDS. Since SynthTabNet is a comprehensive dataset that can thoroughly evaluate the model under different table configurations, we use it for ablations and present results on four subsets separately in Sec. 5.

PubTables-1M [36] aims to overcome the groundtruth inconsistency observed in prior datasets using a new canonicalization procedure. The dataset has 947k tables annotated with bbox and text within each bbox. The dataset differs from previous datasets that the bbox is word-wise instead of cell-wise, thus each cell can have more than one bbox. Besides, since all annotations are in bbox format and no table structure labels, *e.g.*, HTML, are provided, the baseline DETR trained by the dataset creators report their performance in detection metrics, *e.g.*, AP_{50} and AP_{75}. UniTable also achieves competitive results compared with the baseline DETR. Visualizing the bbox predictions shows that our model predicts more bboxes (longer sequence) than the groundtruth, motivating us to delve deep into the dataset annotations and discover previously unacknowledged inconsistencies in table annotations. We conjecture the SOTA performance of DETR model on PubTables-1M may be due to overfitting to the training set. Sec. F.1 describes the details of these dataset annotation issues.

4.4 Using UniTable in Practice

We have also finetuned our UniTable-large across multiple datasets and released a Jupyter Notebook as a demo of our inference pipeline. A user can simply pass a table screenshot through our notebook and obtain a fully digitalized HTML table. The table structure is trained across PubTabNet, FinTabNet, and SynthTabNet, the cell bbox is trained across PubTabNet and SynthTabNet, and the cell content is trained across PubTabNet, SynthTabNet, and PubTables-1M. We have provided a public API hosted on HuggingFace to facilitate the access to our UniTable: `https://poloclub.github.io/magic-table/`. Appendix B visualizes several table examples in practice.

Table 2: Effectiveness and scalability of SSP on all four subsets of SynthTabNet. Without SSP, the model performance suffers, and increasing the model complexity from base to large barely improves the performance. Pretraining the visual encoder on 1M tabular images provides an average increase of 14.40 pp. Pretraining on 2M images continues to increase the performance by 0.74 pp. Here we present S-TEDS of the table structure prediction, and the same trend also applies to other tasks as elaborated in Appendix C.

	Finance		PubTabNet		Marketing		Sparse	
	Base	Large	Base	Large	Base	Large	Base	Large
No SSP	88.95	90.75	89.10	91.67	68.05	70.60	85.50	87.72
SSP 1M	98.73	99.56	99.02	99.55	95.14	99.05	97.20	99.29
SSP 2M	99.41	**99.58**	99.44	**99.56**	98.35	**99.08**	98.69	**99.34**

5 Deeper Dive into UniTable: Ablation and Analysis

5.1 Effectiveness and Scalability of SSP

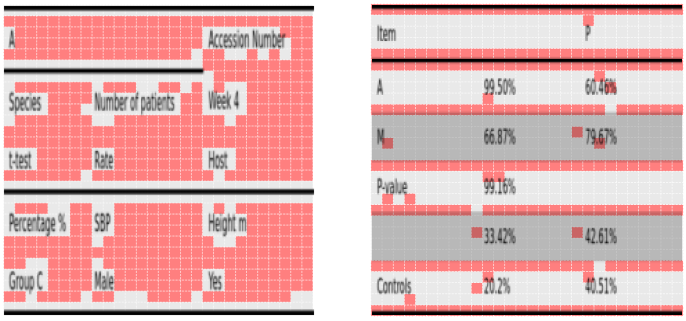
We ablate the effectiveness and scalability of SSP on all four subsets of SynthTabNet since SynthTabNet is a large-scale dataset that can comprehensively evaluate the model under different table configurations. Table 2 presents the table structure task, and the same trend also applies to other tasks as elaborated in Appendix C. Comparing row "No SSP" and "SSP 1M" or "SSP 2M", both the base and the large models have benefited significantly from the SSP. Specifically, Marketing and Sparse are two of the most challenging subsets. Marketing has a colorful background with high-contrast texts, and Sparse contains tables with sparse content. These variations make it challenging to accurately predict the HTML table structure tags and bbox surrounding the cell content. Without SSP, the model performance suffers, and increasing the model complexity from base to large barely improves the performance. Instead, after pretraining the visual encoder on 1M tabular images, both the base and the large models have an average increase of 14.40 pp. Note the performance also scales along with the pretraining dataset size. When the pretraining dataset increases from 1M to 2M, the average performance continues to increase by 0.74 pp. Finally, all large models have an average performance gain of 1.51 pp over the base models.

5.2 Generalization of the Unified Training Objective

Table 3: UniTable's unified training objective applies to both linear projection Transformer and hybrid CNN-Transformer architectures conventionally used in TR. Results on all four subsets of the SynthTabNet for table structure prediction evaluated with S-TEDS, and the same conclusion also applies to other tasks as elaborated in Appendix D.

Model	Finance	PubTabNet	Marketing	Sparse
Base	98.63	98.80	97.16	95.30
Large	99.44	99.44	98.71	98.64

We showcase that our unified training objective, language modeling, not only works on models with linear projection Transformer, but also works with the hybrid CNN-Transformer architecture used in the TR literature. Table 3 presents the results on all four subsets of the SynthTabNet similar to the settings in Sec. 5.1. The main difference is that the linear projection in the visual encoder is replaced by a ResNet-18 CNN backbone. Such a modification leads to an increase of 12M parameters. The performance of this hybrid CNN-Transformer architecture is roughly on par with the linear projection model initialized from the SSP 1M tabular images, which shows that our training objective is agnostic to the choice of architecture. The significant performance gap between hybrid CNN-Transformer and linear projection Transformer trained from scratch verifies the observation identified in the previous literature [30]. Though hybrid CNN-Transformer has also achieved competitive results, we still recommend using the linear projection Transformer because of 1) capability of leveraging the power of SSP, 2) architectural compliance with VLM in natural image domain, and 3) the performance of hybrid CNN-Transformer is still worse than the SSP on 2M tabular images even with more total parameters. Though prior work has started to propose SSP for hybrid CNN-Transformer, this direction is still in its early stage as it is challenging in heavy computational cost and pretraining-finetuning discrepancy [9]. Moreover, most work only demonstrates the capability in discriminative tasks rather



(a) *Background* patches (red) (b) Patches for *separations*

Figure 3: The key reason that SSP works is because each tokens have visual semantics and the codebook shows a fine-grained categorization to represent the implicit conventions in the table. The codebook used in SSP has learned to represent abstract concepts by using different groups of tokens to represent different concepts: (a) empty background and (b) separations within a table. Red highlights the token indices under investigation. Appendix E provides a zoomed-in version of these images labeled with token indices. We lay the color patches over the original tabular image to present the selected token indices from the 2M VQ-VAE.

than the generative tasks used in the TR domain. Besides, existing SOTA VLMs, *e.g.*, BLIP-2 [17] and LLaVA [20], still employ the linear projection and leave the Transformer to learn the interactions between different patches. As we believe the compliance of architecture is an important step towards generic VLMs, our success in replacing the CNN backbone with linear projection and leveraging a task-agnostic language modeling loss is a cornerstone of incorporating TR in modern VLM training.

5.3 Why does SSP work?

During SSP, the visual encoder is trained to fill in the masked tabular image by selecting visual tokens from the VQ-VAE codebook. Thus, we visualize the selected token indices and analyze whether the tokens have visual semantics. Fig. 3 presents the original table overlaid by the selected token indices from the 2M VQ-VAE codebook. We highlight the token indices under investigation in red. Appendix E provides a zoomed-in version of these images labeled with token indices. The input image size is 448×448 and the patch size is 16×16, so each image has 28×28 tokens chosen from the 16384 entries in the codebook. First, we highlight the tokens representing the blank background (Fig 3a), which covers most of the empty space in the table; leaving only texts and separation lines nonhighlited. Then, we highlight the tokens representing the separation lines (Fig 3b), showing that they well-separate the header row, and the shaded and non-shaded rows. Taking a closer look at the token indices in Fig. 5 and 6 in Appendix E, we find the codebook has learned to represent an abstract concept by assigning multiple tokens, where each reflects a different scenario. For example, token "111" represents the separation above the gray color shading and token "964" represents the one below, and token "15282" represents the separation above a bold horizontal line and "10807" represents the one below. A certain type of separation also has multiple choices depending on the portion of the line within the patch, *e.g.*, token "14181", "8714", and "10807" all representing the "below a bold horizontal line", but the differences lie in the line thickness and amount of black inside the image patch. Such a fine-grained categorization also explains why the codebook needs so many tokens to represent the implicit conventions created by humans in the table.

6 Conclusion

We present UniTable, a training framework that unifies both the training paradigm and training objective of TR. Its training paradigm combines the simplicity of purely pixel-level inputs with the effectiveness and scalability empowered by SSP from diverse unannotated tabular images. Our framework unifies the training objectives of all three TR tasks, extracting table structure, cell content, and cell bbox, into a unified task-agnostic training objective: language modeling. Extensive quantitative and qualitative analyses highlights UniTable's SOTA performance on four of the largest TR datasets To promote reproducible research, enhance transparency, and SOTA innovations, we open-source our code and release the first-of-its-kind Jupyter Notebook of the whole inference pipeline, fine-tuned across multiple TR datasets, supporting all three TR tasks.

References

[1] Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, et al. Gpt-4 technical report. *arXiv preprint arXiv:2303.08774*, 2023.

[2] Hangbo Bao, Li Dong, Songhao Piao, and Furu Wei. Beit: Bert pre-training of image transformers. *arXiv preprint arXiv:2106.08254*, 2021.

[3] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[4] Ting Chen, Saurabh Saxena, Lala Li, David J Fleet, and Geoffrey Hinton. Pix2seq: A language modeling framework for object detection. *arXiv preprint arXiv:2109.10852*, 2021.

[5] Ting Chen, Saurabh Saxena, Lala Li, Tsung-Yi Lin, David J Fleet, and Geoffrey E Hinton. A unified sequence interface for vision tasks. *Advances in Neural Information Processing Systems*, 35:31333–31346, 2022.

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

[7] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, et al. An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*, 2020.

[8] Liangcai Gao, Yilun Huang, Hervé Déjean, Jean-Luc Meunier, Qinqin Yan, Yu Fang, Florian Kleber, and Eva Lang. Icdar 2019 competition on table detection and recognition (ctdar). In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1510–1515. IEEE, 2019.

[9] Peng Gao, Teli Ma, Hongsheng Li, Ziyi Lin, Jifeng Dai, and Yu Qiao. Convmae: Masked convolution meets masked autoencoders. *arXiv preprint arXiv:2205.03892*, 2022.

[10] Max Göbel, Tamir Hassan, Ermelinda Oro, and Giorgio Orsi. Icdar 2013 table competition. In *2013 12th International Conference on Document Analysis and Recognition*, pages 1449–1453. IEEE, 2013.

[11] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[12] Yongshuai Huang, Ning Lu, Dapeng Chen, Yibo Li, Zecheng Xie, Shenggao Zhu, Liangcai Gao, and Wei Peng. Improving table structure recognition with visual-alignment sequential coordinate modeling. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 11134–11143, 2023.

[13] Eric Jang, Shixiang Gu, and Ben Poole. Categorical reparameterization with gumbel-softmax. *arXiv preprint arXiv:1611.01144*, 2016.

[14] Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. Ocr-free document understanding transformer. In *European Conference on Computer Vision*, pages 498–517. Springer, 2022.

[15] Kenton Lee, Mandar Joshi, Iulia Raluca Turc, Hexiang Hu, Fangyu Liu, Julian Martin Eisenschlos, Urvashi Khandelwal, Peter Shaw, Ming-Wei Chang, and Kristina Toutanova. Pix2struct: Screenshot parsing as pretraining for visual language understanding. In *International Conference on Machine Learning*, pages 18893–18912. PMLR, 2023.

[16] Junnan Li, Dongxu Li, Caiming Xiong, and Steven Hoi. Blip: Bootstrapping language-image pre-training for unified vision-language understanding and generation. In *International Conference on Machine Learning*, pages 12888–12900. PMLR, 2022.

[17] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. *arXiv preprint arXiv:2301.12597*, 2023.

[18] Minghao Li, Lei Cui, Shaohan Huang, Furu Wei, Ming Zhou, and Zhoujun Li. Tablebank: Table benchmark for image-based table detection and recognition. In *Proceedings of the Twelfth Language Resources and Evaluation Conference*, pages 1918–1925, 2020.

[19] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[20] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *arXiv preprint arXiv:2304.08485*, 2023.

[21] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024.

[22] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 10012–10022, 2021.

[23] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

[24] Ning Lu, Wenwen Yu, Xianbiao Qi, Yihao Chen, Ping Gong, Rong Xiao, and Xiang Bai. Master: Multi-aspect non-local network for scene text recognition. *Pattern Recognition*, 117:107980, 2021.

[25] Maksym Lysak, Ahmed Nassar, Nikolaos Livathinos, Christoph Auer, and Peter Staar. Optimized table tokenization for table structure recognition. *arXiv preprint arXiv:2305.03393*, 2023.

[26] Chixiang Ma, Weihong Lin, Lei Sun, and Qiang Huo. Robust table detection and structure recognition from heterogeneous document images. *Pattern Recognition*, 133:109006, 2023.

[27] David Mizrahi, Roman Bachmann, Oğuzhan Fatih Kar, Teresa Yeo, Mingfei Gao, Afshin Dehghan, and Amir Zamir. 4m: Massively multimodal masked modeling. *arXiv preprint arXiv:2312.06647*, 2023.

[28] Ahmed Nassar, Nikolaos Livathinos, Maksym Lysak, and Peter Staar. Tableformer: Table structure understanding with transformers. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4614–4623, 2022.

[29] OpenAI. Hello gpt-4o, 2024. URL `https://openai.com/index/hello-gpt-4o/`.

[30] Anthony Peng, Seongmin Lee, Xiaojing Wang, Rajarajeswari Raji Balasubramaniyan, and Duen Horng Chau. High-performance transformers for table structure recognition need early convolutions. In *NeurIPS 2023 Second Table Representation Learning Workshop*, 2023.

[31] Devashish Prasad, Ayan Gadpal, Kshitij Kapadni, Manish Visave, and Kavita 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*, pages 572–573, 2020.

[32] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[33] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[34] Huawen Shen, Xiang Gao, Jin Wei, Liang Qiao, Yu Zhou, Qiang Li, and Zhanzhan Cheng. Divide rows and conquer cells: Towards structure recognition for large tables. In *Proceedings of the Thirty-Second International Joint Conference on Artificial Intelligence*, pages 1369–1377, 2023.

[35] Yongxin Shi, Dezhi Peng, Wenhui Liao, Zening Lin, Xinhong Chen, Chongyu Liu, Yuyi Zhang, and Lianwen Jin. Exploring ocr capabilities of gpt-4v (ision): A quantitative and in-depth evaluation. *arXiv preprint arXiv:2310.16809*, 2023.

[36] Brandon Smock, Rohith Pesala, and Robin Abraham. Pubtables-1m: Towards comprehensive table extraction from unstructured documents. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4634–4642, 2022.

[37] Aaron Van Den Oord, Oriol Vinyals, et al. Neural discrete representation learning. *Advances in neural information processing systems*, 30, 2017.

[38] Yonghui Wu, Mike Schuster, Zhifeng Chen, Quoc V Le, Mohammad Norouzi, Wolfgang Macherey, Maxim Krikun, Yuan Cao, Qin Gao, Klaus Macherey, et al. Google's neural machine translation system: Bridging the gap between human and machine translation. *arXiv preprint arXiv:1609.08144*, 2016.

[39] Zhengyuan Yang, Linjie Li, Kevin Lin, Jianfeng Wang, Chung-Ching Lin, Zicheng Liu, and Lijuan Wang. The dawn of lmms: Preliminary explorations with gpt-4v (ision). *arXiv preprint arXiv:2309.17421*, 9(1):1, 2023.

[40] Jiaquan Ye, Xianbiao Qi, Yelin He, Yihao Chen, Dengyi Gu, Peng Gao, and Rong Xiao. Pingan-vcgroup's solution for icdar 2021 competition on scientific literature parsing task b: table recognition to html. *arXiv preprint arXiv:2105.01848*, 2021.

[41] Xinyi Zheng, Douglas Burdick, Lucian Popa, Xu Zhong, and Nancy Xin Ru Wang. Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context. In *Proceedings of the IEEE/CVF winter conference on applications of computer vision*, pages 697–706, 2021.

[42] Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. Image-based table recognition: data, model, and evaluation. In *European conference on computer vision*, pages 564–580. Springer, 2020.

A Detail comparisons with milestone VLMs

Table 4: UniTable outperforms GPT-4V and LLaVA-v1.6 by a huge margin for table recognition.

Model	Finance			PubTabNet			Marketing			Sparse		
Samples	100	300	500	100	300	500	100	300	500	100	300	500
LLaVA-v1.6-vicuna-7b	37.72	35.57	34.16	41.66	42.19	41.80	36.54	34.34	34.14	36.95	37.42	38.49
LLaVA-v1.6-mistral-7b	38.40	39.76	38.07	39.16	37.79	38.96	34.16	33.20	32.25	40.28	35.97	37.06
LLaVA-v1.6-vicuna-13b	43.50	45.13	43.95	48.17	49.05	49.68	41.37	42.00	41.40	46.38	46.04	47.21
LLaVA-v1.6-34b	40.86	42.96	40.69	46.30	47.99	49.30	35.08	35.75	33.99	39.13	39.60	39.85
GPT-4-1106-vision-pre.	64.05	66.93	66.13	69.26	69.86	69.62	58.51	60.03	59.13	55.64	57.02	56.99
UniTable Large	**99.63**	**99.56**	**99.57**	**99.54**	**99.54**	**99.55**	**99.11**	**99.10**	**99.10**	**99.34**	**99.35**	**99.34**

Table 4 shows detailed comparisons (S-TEDS score) with GPT-4V and the latest LLaVA v1.6. In summary, UniTable outperforms GPT-4V and LLaVA-v1.6 by a huge margin for table recognition.

Dataset: SynthTabNet [28], is a large-scale table benchmark that offers control over dataset size, table structure, style, and content type. It has 4 subsets: Finance, PubTabNet, Marketing, Sparse. We incrementally evaluate more samples up to 500 on each subset because no further notable changes in the results are observed.

Models: We compare UniTable with GPT-4V and LLaVA. For GPT-4V, we use the gpt-4-1106-vision-preview. For LLaVA, we test on all 4 variants of its latest version, v1.6, provided by the official model weights on HuggingFace [21]: vicuna-7b, mistral-7b, vicuna-13b, 34b. We use the same prompt as in the latest work that evaluates GPT-4V's OCR capabilities by Shi *et al*., [35]: "Please read the table in this image and return an HTML-style reconstructed table in text, do not omit anything." The max generated sequence length is set to 1500 to cover the largest table in the test.

B Using UniTable in Practice

A user can simply pass a table screenshot through our notebook and obtain a fully digitalized HTML table. We visualize different types of table cell bbox detection results of UniTable in Fig. 4, and an example of a digitalized HTML table can be found in `https://anonymous.4open.science/r/icml-review/notebooks/full_pipeline.ipynb`.

C Effectiveness and Scalability of SSP

Table 10 demonstrates the effectiveness and scalability of SSP on all four subsets of SynthTabNet. Without SSP, the model performance suffers, and increasing the model complexity from base to large barely improves the performance. Pretraining the visual encoder on 1M tabular images provides a significant improvement, and pretraining on 2M images continues to increase the performance on all TR tasks.

D Generalization of the Unified Training Objective

UniTable's unified training objective applies to both linear projection Transformer and hybrid CNN-Transformer architectures conventionally used in TR. Table 11 shows results on all four subsets of the SynthTabNet for table structure prediction evaluated with S-TEDS and table cell bbox detection evaluated with mAP.

E Visualization of the Visual Codebook

Fig. 5 and 6 present the original table overlaid by the selected token indices from the 2M VQ-VAE codebook. We highlight the token indices under investigation in red and use blue for the others. The input image size is 448×448 and the patch size is 16×16, so each image has 28×28 tokens chosen from the 16384 entries in the codebook. First, we highlight the tokens representing the

	IC19B2M		PubTabNet			FinTabNet	SynthTabNet		PubTables-1M	
	IoU 0.6	WAvg. F1	AP_{50}	S-TEDS	TEDS	S-TEDS	AP_{50}	S-TEDS	AP_{50}	AP_{75}
SOTA	GTE	GTE	VAST	VAST	VAST	VAST	TableFormer	DRCC	DETR	DETR
	38.50	24.80	94.50	97.23	96.31	98.63	87.70	98.70	**97.10**	**94.80**
UniTable										
Base	54.97	40.15	97.94	95.63	94.78	97.19	98.99	98.97	94.48	88.64
Large	**58.10**	**42.62**	**98.43**	**97.89**	**96.50**	**98.89**	**99.00**	**99.39**	95.68	93.28

(a) Table cell bbox detections results on Table 1 in the main paper.

Datasets	Split	GraphConsis	CAREGNN	PC-GNN	BWGNN	MLP	GT	ET (Ours)
Yelp	1%	$56.8_{\pm 2.8}$	$62.1_{\pm 1.3}$	$59.8_{\pm 1.4}$	$61.1_{\pm 0.4}$	$53.9_{\pm 0.2}$	$61.7_{\pm 0.4}$	$\mathbf{63.0}_{\pm \mathbf{0.6}}$
	40%	$58.7_{\pm 2.0}$	$63.3_{\pm 0.9}$	$63.0_{\pm 2.3}$	$71.0_{\pm 0.9}$	$57.5_{\pm 0.8}$	$68.7_{\pm 0.4}$	$\mathbf{71.5}_{\pm \mathbf{0.1}}$
Amazon	1%	$68.5_{\pm 3.4}$	$68.7_{\pm 1.6}$	$79.8_{\pm 5.6}$	$\mathbf{90.9}_{\pm \mathbf{0.7}}$	$74.6_{\pm 1.2}$	$88.6_{\pm 0.5}$	$89.3_{\pm 0.7}$
	40%	$75.1_{\pm 3.2}$	$86.3_{\pm 1.7}$	$89.5_{\pm 0.7}$	$92.2_{\pm 0.4}$	$79.1_{\pm 1.2}$	$91.7_{\pm 0.8}$	$\mathbf{92.8}_{\pm \mathbf{0.3}}$
T-Finance	1%	71.7	73.3	62.0	84.8	61.0	81.5	$\mathbf{85.1}_{\pm \mathbf{1.0}}$
	40%	73.4	77.5	63.1	86.8	70.5	83.6	$\mathbf{88.2}_{\pm \mathbf{1.0}}$
T-Social	1%	52.4	55.8	51.1	75.9	50.0	64.3	$\mathbf{79.1}_{\pm \mathbf{0.7}}$
	40%	56.5	56.2	52.1	**83.9**	50.3	68.2	$83.5_{\pm 0.4}$
Yelp	1%	$66.4_{\pm 3.4}$	$75.0_{\pm 3.8}$	$\mathbf{75.4}_{\pm \mathbf{0.9}}$	$72.0_{\pm 0.5}$	$59.8_{\pm 0.4}$	$72.5_{\pm 0.6}$	$73.2_{\pm 0.8}$
	40%	$69.8_{\pm 3.0}$	$76.1_{\pm 2.9}$	$79.8_{\pm 0.1}$	$84.0_{\pm 0.9}$	$66.5_{\pm 1.0}$	$81.9_{\pm 0.5}$	$\mathbf{84.9}_{\pm \mathbf{0.3}}$
Amazon	1%	$74.1_{\pm 3.5}$	$88.6_{\pm 3.5}$	$90.4_{\pm 2.0}$	$89.4_{\pm 0.3}$	$83.6_{\pm 1.7}$	$89.0_{\pm 1.2}$	$\mathbf{91.9}_{\pm \mathbf{1.0}}$
	40%	$87.4_{\pm 3.3}$	$90.5_{\pm 1.6}$	$95.8_{\pm 0.1}$	$\mathbf{98.0}_{\pm \mathbf{0.4}}$	$89.8_{\pm 1.0}$	$95.4_{\pm 0.6}$	$97.3_{\pm 0.4}$
T-Finance	1%	90.2	90.5	90.7	91.1	82.9	90.0	$\mathbf{92.8}_{\pm \mathbf{1.1}}$
	40%	91.4	92.1	91.2	94.3	87.1	88.2	$\mathbf{95.0}_{\pm \mathbf{3.0}}$
T-Social	1%	65.2	71.2	59.8	88.0	56.3	81.4	$\mathbf{91.9}_{\pm \mathbf{0.6}}$
	40%	71.2	71.8	68.4	**95.2**	56.9	82.5	$93.9_{\pm 0.2}$

(b) Table cell bbox detections results on complex academic tables

Medical Plans		
Health Alliance Plan HMO Group #: 10000664	800-422-4641 (Mon–Fri 8am–7pm)	web: hap.org app: HAP OnTheGo
Priority Health HMO Group #: 796653	800-446-5674	web: priorityhealth.com app: Priority Health Member Portal
Blue Care Network HMO Group #: 00111308	800-662-6667 (Mon–Fri 8am–5:30pm)	web: bcbsm.com app: BCBSM
Community Blue PPO Group #: 007002779	877-354-2583 (Mon–Fri 8am–5:30pm)	web: bcbsm.com app: BCBSM
Blue Cross Blue Shield of Michigan Group #: 007002779	877-354-2583 (Mon–Fri 8am–5:30pm)	web: bcbsm.com app: BCBSM
Virtual Doctor Visits (visit a board-certified doctor via smartphone or computer 24/7)		
HAP – American Well	844-733-3627 (every day, 24 hours)	web: hap.amwell.com email: support@amwell.com app: Amwell: Doctor Visits 24/7 **Service Key:** HAPMi
Priority Health – Spectrum Health Now	844-322-7374	web: priorityhealth.com app: Spectrum Health
Blue Cross Online Visits for: Blue Care Network, Community Blue & Blue Cross Blue Shield	844-606-1608 (every day, 24 hours)	web: bcbsmonlinevisits.com app: BCBSM Online Visits

(c) Table cell bbox detections results on tables with colorful backgrounds and spanning headers.

Figure 4: Table cell bbox detection results of UniTable on unannotated tables in practice.

blank white background. Comparing the red and blue indices, we observe that the red region has covered most of the blank space inside the table, and the blue region has formulated a convex hull tightly surrounding either the texts or separation lines. Next, we highlight the tokens representing the separation lines. Comparing the red and blue regions, we observe that the red regions align with all the separations within the table, *e.g.*, line separations and color shading separations. Taking a closer look at the indices, we find the codebook has learned to represent an abstract concept by assigning multiple tokens, where each reflects a different scenario. For example, token "111" represents the separation above the gray color shading and token "964" represents the one below, and token "15282" represents the separation above a bold horizontal line and "10807" represents the one below. A certain type of separation also has multiple choices depending on the portion of the line within the patch, *e.g.*, token "14181", "8714", and "10807" all representing the "below a bold horizontal line", but the differences lie in the line thickness and amount of black inside the image patch. Such a fine-grained categorization also explains why the codebook needs so many tokens to represent the implicit conventions created by humans in the table.

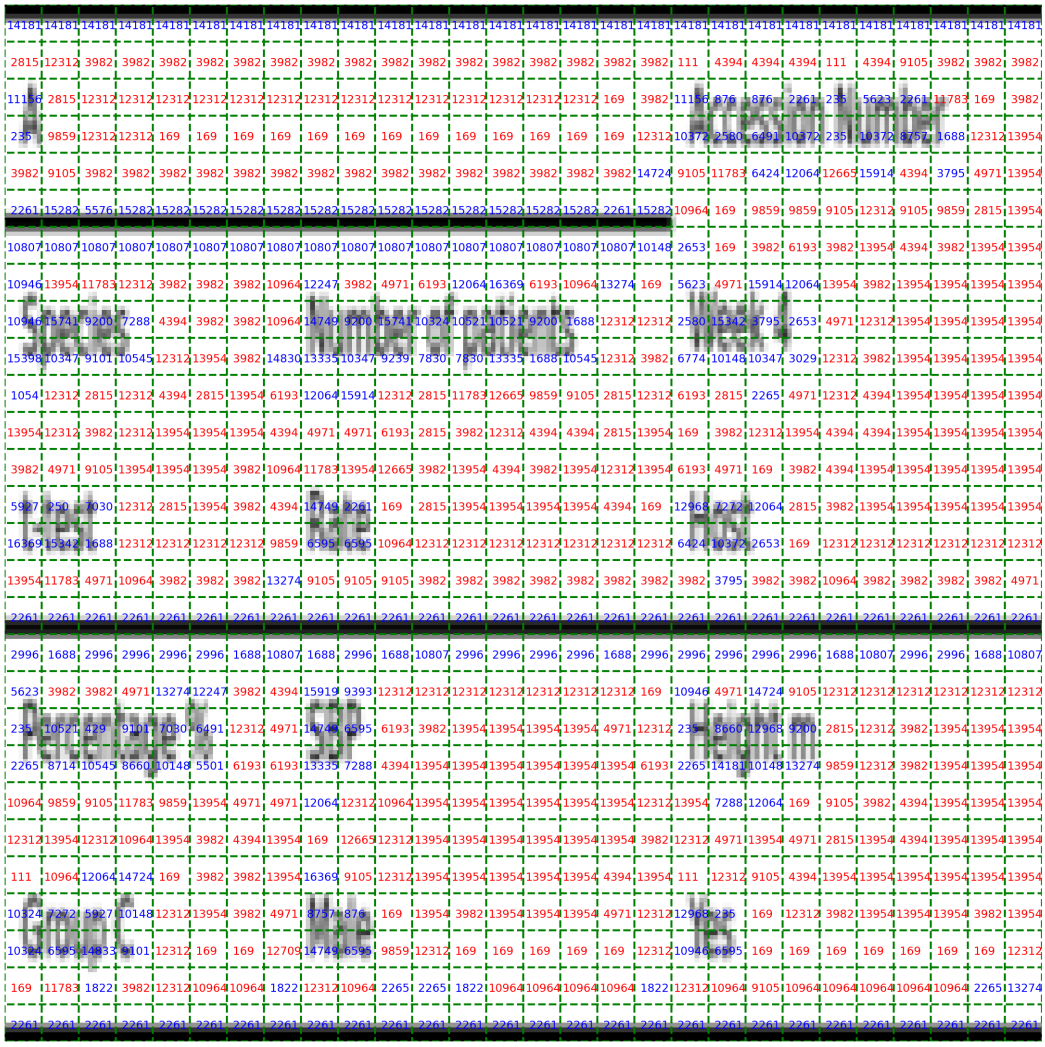


Figure 5: A zoomed-in version of the token indices from the 2M VQ-VAE. The codebook used in SSP has learned to represent abstract concepts by using different groups of tokens to represent empty white backgrounds within a table. Red highlights the token indices representing the concept of "empty white backgrounds".

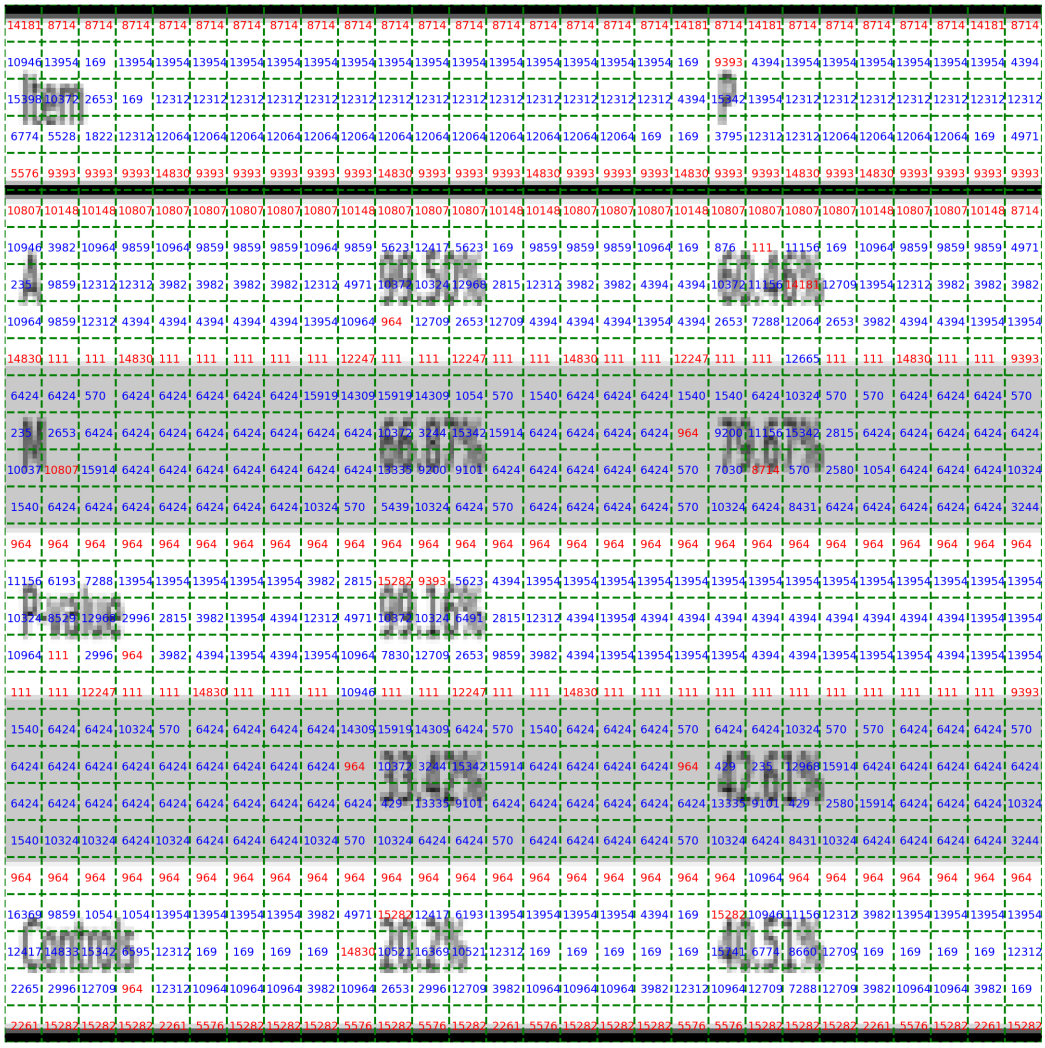


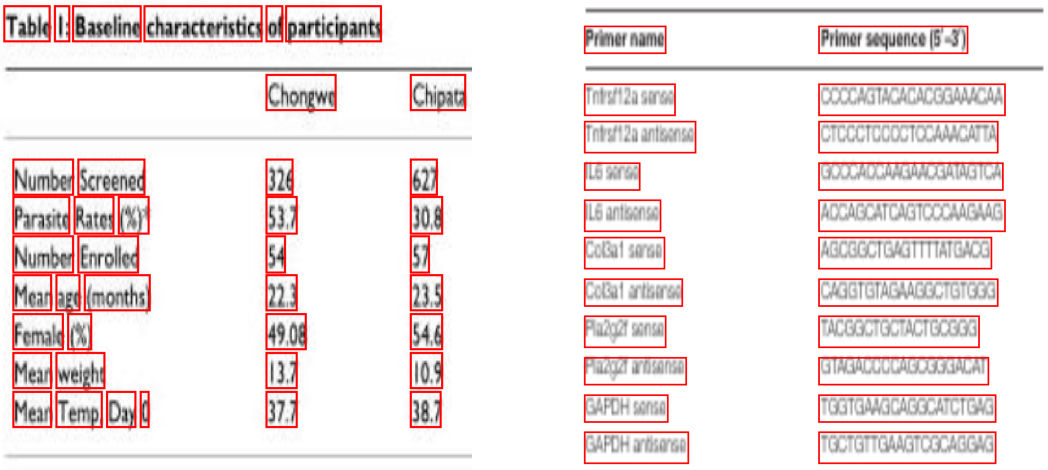
Figure 6: A zoomed-in version of the token indices from the 2M VQ-VAE. The codebook used in SSP has learned to represent abstract concepts by using different groups of tokens to represent separations within a table. Red highlights the token indices representing the concept of "separations".

F Discussions on TSR Dataset Annotations

Word-wise vs. cell-wise bbox annotations. Fig. 7a is the word-wise annotation from table "PMC1574335 table 0" in PubTables-1M and Fig. 7b is the cell-wise annotation from table "PMC5897438 004 00" in PubTabNet. In cell-wise annotation, each cell has a unique bbox that can be matched to the non-empty cells in table structure HTML tags. However, in word-wise annotation, a bbox is matched to a single word, which cannot be easily combined with the table structure as in cell-wise annotation, which limits its general applicability.

F.1 Discussions on TSR Dataset Annotations

Our goal is to push forward the development of the document and table understanding, thus we hope to see high-quality and consistent data annotations. Because of the unified language modeling framework formulation, we discover previously unacknowledged inconsistency in one of the largest TR datasets, PubTables-1M.



(a) Word-wise bbox from PubTables-1M

(b) Cell-wise bbox from PubTabNet

Figure 7: Word-wise vs. cell-wise bbox annotations: (a) is the word-wise annotation from table "PMC1574335 table 0" in PubTables-1M and (b) is the cell-wise annotation from table "PMC5897438 004 00" in PubTabNet. In cell-wise annotation, each cell has a unique bbox that can be matched to the non-empty cells in table structure HTML tags. However, in word-wise annotation, a bbox is matched to a single word, which cannot be easily combined with the table structure as in cell-wise annotation, which limits its general applicability.

(a) Unrelated text around table

(b) Overlapping bbox

Figure 8: Our UniTable has successfully identified inconsistent table cell bbox annotations in PubTables-1M, which can guide future data annotation pipelines and qualities.

Here, we have identified three types of inconsistent table cell bbox annotations: (1) Fig. 8a ("PMC4964067 table 7") shows bbox annotations for unrelated text around table; (2) Fig. 8b ("PMC6202420 table 1") shows overlapping bbox annotations; (3) Bbox annotations exceed the boundary of image size, *e.g.*, `[-4.6, 278.6, 19.5, 292.4]` from "PMC4802837 table 1". With a rapid filter comparing the image size with the bbox, we find that a surprising 53.10% (402914 over 758849) of the table annotations in the training set have at least one bbox going beyond the image size. We hope that these inconsistencies discovered by UniTable can help guide future data annotation pipelines and qualities.

G SOTA results

Table 5: Comparisons on IC19B2M

Model	IoU 0.6	WAvg. F1
NLPR-PAL [8]	30.50	20.60
CascadeTabNet [31]	35.40	23.20
GTE [41]	38.50	24.80
UniTable Base (Ours)	54.97	40.15
UniTable Large (Ours)	**58.10**	**42.62**

Table 6: Comparisons on PubTabNet

Model	S-TEDS	TEDS	AP_{50}	AP_{75}
EDD [42]	89.90	88.30	79.20	-
GTE [41]	93.01	-	-	-
TableFormer [28]	96.75	93.60	82.10	-
TableMaster [40]	96.04	96.16	-	-
OTSL [25]	95.50	-	-	88.00
VAST [12]	97.23	96.31	94.80	-
UniTable Base (Ours)	95.63	94.78	97.94	87.27
UniTable Large (Ours)	**97.89**	**96.50**	**98.43**	**92.44**

Table 7: Comparisons on SynthTabNet

Model	S-TEDS	AP_{50}	AP_{75}
TableFormer [28]	96.70	87.70	-
DRCC [34]	98.70	-	-
UniTable Base (Ours)	98.97	98.99	98.79
UniTable Large (Ours)	**99.39**	**99.00**	**98.87**

Table 8: Comparisons on FinTabNet

Model	S-TEDS
GTE [41]	91.02
EDD [42]	90.60
OTSL [25]	95.90
TableFormer [28]	96.80
VAST [12]	98.63
UniTable Base (Ours)	97.19
UniTable Large (Ours)	**98.89**

Table 5, 6, 7, 8, and 9 show detailed comparisons across five of the largest table benchmarks from our paper. In summary, UniTable outperforms prior methods and achieves SOTA on four out of the five largest table datasets. ("-" means not reported by the compared methods.)

Table 9: Comparisons on PubTables-1M

Model	AP_{50}	AP_{75}
Faster R-CNN [36]	81.50	78.50
OTSL [25]	-	89.60
DETR [36]	**97.10**	**94.80**
UniTable Base (Ours)	94.48	88.64
UniTable Large (Ours)	95.68	93.28

Table 10: Effectiveness and scalability of SSP on all four subsets of SynthTabNet. Without SSP, the model performance suffers, and increasing the model complexity from base to large barely improves the performance. Here we present both S-TEDS of the table structure prediction and mAP of the cell bbox detection

	Finance		PubTabNet		Marketing		Sparse	
	Base	Large	Base	Large	Base	Large	Base	Large
Table structure - S-TEDS								
No SSP	88.95	90.75	89.10	91.67	68.05	70.60	85.50	87.72
SSP 1M	98.73	99.56	99.02	99.55	95.14	99.05	97.20	99.29
SSP 2M	99.41	**99.58**	99.44	**99.56**	98.35	**99.08**	98.69	**99.34**
Table cell bbox - COCO mAP								
No SSP	83.30	85.46	87.67	88.88	72.00	75.21	88.84	90.01
SSP 1M	96.13	97.07	95.93	96.95	94.63	95.91	97.07	97.84
SSP 2M	96.54	**97.37**	96.50	**97.44**	95.21	**96.25**	97.46	**97.96**

Table 11: UniTable's unified training objective applies to both linear projection Transformer and hybrid CNN-Transformer architectures conventionally used in TR. Results on all four subsets of the SynthTabNet for table structure prediction evaluated with S-TEDS and cell bbox detection with mAP.

Model	Finance	PubTabNet	Marketing	Sparse
Table structure - S-TEDS				
Base	98.63	98.80	97.16	95.30
Large	99.44	99.44	98.71	98.64
Table cell bbox - COCO mAP				
Base	94.61	94.39	91.42	95.38
Large	95.90	96.50	94.96	97.55