Tabularis Revilio: Converting Text to Tables

Mukul Singh
Microsoft
Redmond, USA
singhmukul@microsoft.com

Sumit Gulwani
Microsoft
Redmond, USA
sumitg@microsoft.com

Vu Le
Microsoft
Redmond, USA
levu@microsoft.com

Gust Verbruggen
Microsoft
Keerbergen, Belgium
gverbruggen@microsoft.com

ABSTRACT

Copying tables from documents and applications without proper tabular support, like PDF documents, web pages or images, surprisingly remains a challenge. In this paper, we present Revilio, a novel neurosymbolic system for reconstructing tables when their column boundaries have been lost. Revilio addresses this task by detecting headers, generating an initial table sketch using a large language model, and using that sketch as a guiding representation during an enumerate-and-test strategy that evaluates syntactic and semantic table structures. We evaluate Revilio on a diverse set of datasets, demonstrating significant improvements over existing table parsing methods. Revilio outperforms traditional techniques in both accuracy and scalability, handling large tables with over 100,000 rows. Our experiments find an increase in reconstruction accuracy by 5.8–11.3% over both neural and symbolic baseline systems.

CCS CONCEPTS

• **Information systems** → *Document structure*; **Information extraction**; • **Computing methodologies** → **Information extraction**; • **Applied computing** → *Document management*.

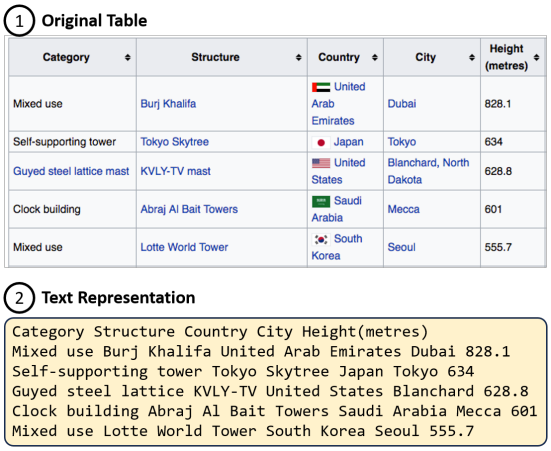
KEYWORDS

Table Construction, Data Extraction, Language Models for Tables

ACM Reference Format:
Mukul Singh, Sumit Gulwani, Vu Le, and Gust Verbruggen. 2024. *Tabularis Revilio*: Converting Text to Tables. In *Proceedings of the 33rd ACM International Conference on Information and Knowledge Management (CIKM '24), October 21–25, 2024, Boise, ID, USA.* ACM, New York, NY, USA, 5 pages.
https://doi.org/10.1145/3627673.3680000

1 INTRODUCTION

Tables are commonly used to store and present data. Surprisingly, these tables are often moved as free-form text, for example, when copying tables from rendered documents like PDF and websites.



① **Original Table**

Category	Structure	Country	City	Height (metres)
Mixed use	Burj Khalifa	United Arab Emirates	Dubai	828.1
Self-supporting tower	Tokyo Skytree	Japan	Tokyo	634
Guyed steel lattice mast	KVLY-TV mast	United States	Blanchard, North Dakota	628.8
Clock building	Abraj Al Bait Towers	Saudi Arabia	Mecca	601
Mixed use	Lotte World Tower	South Korea	Seoul	555.7

② **Text Representation**

```
Category Structure Country City Height(metres)
Mixed use Burj Khalifa United Arab Emirates Dubai 828.1
Self-supporting tower Tokyo Skytree Japan Tokyo 634
Guyed steel lattice KVLY-TV United States Blanchard 628.8
Clock building Abraj Al Bait Towers Saudi Arabia Mecca 601
Mixed use Lotte World Tower South Korea Seoul 555.7
```

Figure 1: Example showing the text representation of tables. (1) Original table on tallest towers in the world; (2) Text representation of the table without any structural information.

Users are then dependent on manual effort or programming abilities to parse this free-form text back into structured tables.

We introduce Revilio, a system that leverages large language models (LLMs) for table construction from free-form text. Unlike previous methods, which primarily rely on syntactic cues, Revilio uses the semantic knowledge of the language model to ensure that the reconstructed tables are both accurate and natural.

Converting text to tables poses several challenges concerning the semantics and consistency of the table, as well as its scale. Revilio leverages an LLM to detect headers and to build a table sketch from a small subset of the data. Through multi-step reasoning, the language model ensures consistent cell boundaries over a natural granularity. Given this table sketch, Revilio then explores and ranks potential cell boundaries for all the remaining rows. This ranker combines both syntactic and semantic information (based on cell value embeddings) to evaluate consistency of cells across the entire columns, and alignment of rows to the table sketch.

We evaluate Revilio against neural (prompted and fine-tuned) and symbolic baselines on three different datasets, and show an increase in reconstruction accuracy of 5.8–11.3% compared to these baselines. We show that it is able to handle large tables (> 100K rows) and analyze the impact of our design decisions on this performance.

In short, we make the following contributions:

CIKM '24, October 21–25, 2024, Boise, ID, USA

ACM ISBN 979-8-4007-0436-9/24/10
https://doi.org/10.1145/3627673.3680000

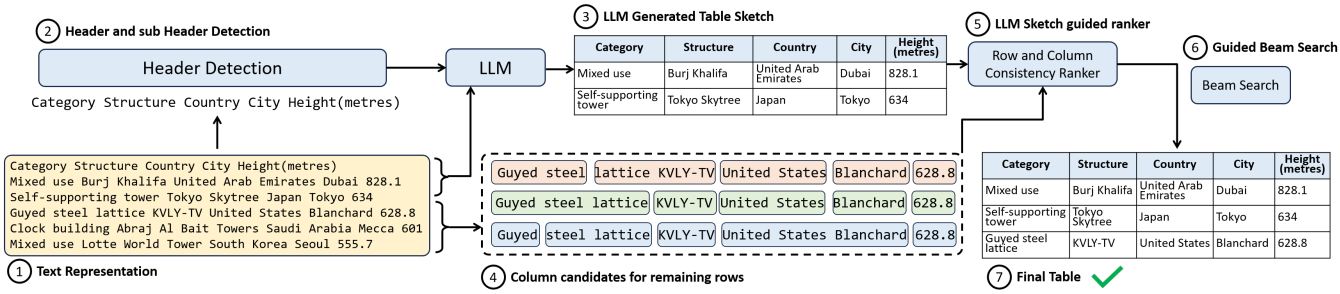


Figure 2: Summary of Revilio on a sample table. (1) A table with 100+ rows; (2) Revilio first performs header and subheader detection to understand the table structure; (3) The header and 2 more sample rows are then prompted to the LLM to generate a preliminary table sketch; (4) The remaining rows are split based on common delimiters to generate candidate column splits; (5) The LLM generated sketch is used to rank the split based on column and row level consistency of the resulting table; (6) The ranker is used to perform a guided beam search over the row candidates to generate the final table (7).

- We combine the semantic knowledge of LLMs with the speed of a generate-and-rank strategy to accurately and quickly convert text to (large) tables.
- We evaluate Revilio on three datasets and show that it achieves higher table reconstruction accuracy at a fraction of the cost of different neural and symbolic baselines.

2 RELATED WORK

The detection of tables from various formats has been extensively studied, particularly in the context of images, PDF, and web data.

Text to table. Generating tables from textual descriptions and online sources is another active research area. This involves transforming unstructured text into structured tabular formats. Techniques in this domain focus on understanding natural language descriptions and generating corresponding tables. Methods like InfoTab [9], Text2Table [1] and Text-to-Table [28] employ natural language processing (NLP) and template-based approaches to map textual content to table structures. However, these methods often struggle with diverse and unstructured text inputs.

Web to table. The detection of tables from web pages leverages the HTML structure and CSS styles to identify and extract tables directly from the source. Early approaches used rule-based systems to parse HTML tags [7], while recent methods employ dedicated models to enhance accuracy and generalization [24].

Image to table. Early work focused on extracting tables from scanned images use computer vision techniques. These approaches typically involve detecting table boundaries and cell divisions using edge detection and contour analysis [18, 23, 26] or (convolutional) neural neural networks to improve the accuracy and robustness of table detection from images [17, 20, 22]. Extracting tables from OCR text and PDF documents presents unique challenges due to the lack of explicit structural information. Methods such as Tabula [29] and Camelot [3] employ heuristic-based approaches to detect table structures by analyzing spatial relationships between text blocks. More recent techniques utilize neural models to classify text blocks and infer table structures [5, 6, 11]. These methods struggle with complex tables that lack boundaries or contain nested structures.

3 PRELIMINARIES

Let $\mathbf{T} = [t_i^j]_{i=1\to n}^{j=1\to m}$ be a table with n rows and m columns. We write T_i and T^j to denote row i and column j, respectively. The header $\mathbf{H} = [h_i^j]_{i=1\to n}^{j=1\to k}$ of a table are special cells that define the semantic structure of the table by assigning one or more names H_i to each column i. A text-form table $\mathbf{R}$ is a deconstructed representation of a table $\mathbf{T}$ obtained by concatenating the values in each row (including headers) with a single space character to obtain $m + k$ text rows R_i. An example of a table and its text form are shown in Figure 1. In this work, we tackle the task of converting a text-form table $\mathbf{R}$ to a structured table $\mathbf{T}$ without loss of information, and where each header H_i correctly describes the data in its column.

4 METHOD

Figure 2 summarizes the architecture of Revilio with an example. The example shown is one where GPT-4 (best baseline) fails. The following sections describe each component in more detail.

4.1 Detecting headers

We train a simple embedding based classifier to predict if a row R_i is header or not. The input to the model is the embedding of the cells in the row. We use SentenceBERT [19]. If no rows are classified as header then Revilio assumes that the table does not have headers.

4.2 Generating table sketches

In this step, we leverage a single LLM prompt to generate initial sketch $T_\mathbf{d}$. We use an off-the-shelf string profiler FlashProfile [16] to learn a pattern for each row and use these patterns to sample the five most diverse rows $R_\mathbf{d}$. The header rows (if any) and these five rows are used as input to the LLM. We use a three-step reasoning prompt and instruct the model to describe the number of rows and columns, the header, and the final table. For the table, we use a program-of-thought prompt [4] where the model is instructed to generate a pandas DataFrame using comments and assertions to guide its thoughts. We can then use the answers to these reasoning step to validate the predicted table: the number of rows, columns and the header are all expected to align. We obtain five completions from the model and select the first one that satisfies the validation.

Table 1: Properties of benchmark tasks for different sources. We report the number of tasks (# Tasks), average number of rows (# Rows), columns (# Columns), and headers (# Headers).

Benchmark	# Tasks	# Rows	# Columns	# Headers
Wikipedia Tables	1000	84.7	8.3	1.2
PubTabNet	1000	14.5	4.9	1.6
CleverCSV	1000	45131.4	87.0	1.1
Total	**3000**	**15076.8**	**33.4**	**1.3**

4.3 Enumerating candidate rows

For each row $R_i \in \mathbf{R} \setminus R_\mathbf{d}$ we enumerate candidate T'_i by identifying $m-1$ splits. We use 4 heuristics for this: (1) analyzing delimiters the LLM assumed to generate $T_\mathbf{d}$ from $R_\mathbf{d}$, (2) non-alphanumeric characters, (3) letter to number change, (4) case change.

4.4 Ranking candidate rows

We compute consistency $\delta(T'_i, \mathbf{T})$ of a new row T'_i with respect to a current table $\mathbf{T}$ by combining one symbolic and two neural metrics. The symbolic metric involves learning a pattern over columns (again using FlashProfile) and computing a weighted (by pattern frequency) pattern edit distance deviation over the column. The neural metrics compute the average embedding similarity within each column, as well as over whole rows. These three metrics are aggregated as a linear combination with learned weights.

4.5 Reconstructing tables

Starting from the sketch $T_\mathbf{d}$, for each row T_i, we greedily select the candidate T'_i that maximizes the consistency $\delta(T'_i, T_\mathbf{d} \cup T'_{<i})$ over $T_\mathbf{d}$ and the selected candidates for all previous rows. Starting from this seed table, Revilio picks the row with the lowest consistency, and performs a beam search (width 5) over alternative candidates for each row until the consistency does not increase for a few iterations (3) or a maximum number of iterations have been reached (100).

4.6 Training data

We train the header detector and neural ranker on the CSV dataset introduced with CleverCSV [25]. This dataset contains 100K noisy CSV files from data.gov.uk and github.com that we parse into 95K clean CSV files using CleverCSV. We further augment this dataset by shuffling columns. To train the ranker, we use the candidate row enumeration (Section 4.3) and sample one wrong candidate (– example) per row (+ example).

5 EVALUATION SETUP

In this section we describe the benchmarks, metrics and baselines.

5.1 Benchmarks

To evaluate Revilio, we consider benchmarks from a diverse set of sources. Table 1 summarizes properties of these datasets.

(1) **Wikipedia Tables**: We use the WikiTables dataset [2] which contains tables from Wikipedia. These are usually short tables with rich formatting and structure with significant semantic content. We generate the text-form representation by manually concatenating values with whitespaces. We sample 1000 tables to create benchmark tasks.
(2) **PubTabNet**: Since Optical Character Recognition (OCR) is a huge area for tables, we use the image table recognition dataset PubTabNet [30] which contains tables found in scientific open source articles along with their OCR annotation. We sample 1000 tables from this and create benchmark tasks by considering the OCR text as the text-form table.
(3) **CleverCSV** [25]: This is a dataset of noisy CSV tables. We sample 1000 tables that were held out from the training set and use the noisy version of the table as input.

5.2 Metrics

We use three metrics to evaluate Revilio. (1) **Table match**: A table is considered exactly reconstructed when all values in the table matches ground truth values. (2) **Column match**: Average percentage of columns that are exactly matched, micro-averaged per table and then averaged across tables. (3) **Value match**: Average percentage of values correctly reconstructed across all tasks.

5.3 Baselines

We compare Revilio to a set of diverse symbolic and neural systems dedicated to table recognition and parsing, and also adapt other popular language and tabular domain techniques on this task.

Symbolic TableLabs is an interactive tool to extract tables from PDFs and raw text. TableLabs detects tables with similar structures (templates) by clustering embeddings from the extraction model. Since, our task does not have user feedback, we do not allow TableLabs to iterate with human feedback.

Language models We fine-tune CodeT5+ [27], StarCoder [12], CodeLlama [21] and Phi-2 [8] on text-to-text objectives where the model has to output valid CSV. Each model is pre-trained on delimiter reconstruction—where 25% of delimiters in a row are removed—and fine-tuned on partial table reconstruction (first three rows + k random rows) $\rightarrow$ table with $k \in \{1, 5, 10, \ldots, 50\}$.

Vision techniques We use image-to-table approaches by converting all text-form tables to images using `matplotlib`. We also consider multi-modal techniques, MuTabNet [10] which achieves SOTA results on 2 of the 4 image-to-text benchmarks. MuTabNet uses a multi-layer cross-attention architecture for table structure detection and OCR mapping. We also use GPT-4V [15] (prompted) and Llava [13, 14] (fine-tuned) for this task as they are SOTA in other visual table tasks for inference and fine tuned respectively.

6 RESULTS

We perform experiments to answer the following research questions. **RQ1:** How accurately does Revilio reconstruct tables compared to baselines? **RQ2:** How well does Revilio handle large tables? **RQ3:** What is the impact of different components of Revilio?

6.1 Performance

Table 2 shows the table, column and value match of Revilio and other baseline systems on reconstructed tables. We find that Revilio outperforms all baselines across all metrics. We find that

Table 2: Comparison of Revilio and baselines on three benchmark datasets using table, column and value match metrics. Technique indicates the base architecture type of the system. Revilio outperforms baselines across all benchmarks.

System description		Wikipedia			PubTabNet			Broken CSV		
Name	**Technique**	**Table**	**Column**	**Value**	**Table**	**Column**	**Value**	**Table**	**Column**	**Value**
TableLabs	Symbolic	50.4	73.2	80.1	50.8	66.5	78.2	40.0	62.2	75.3
CodeT5+ 16B	Language	57.1	81.7	85.6	57.8	73.6	84.1	47.1	67.1	79.2
StarCoder-15B	Language	52.4	76.0	83.6	56.8	71.2	82.4	46.8	65.3	75.4
CodeLlama-13B	Language	52.8	74.0	82.1	55.6	69.3	80.5	44.3	64.1	74.2
GPT-4	Language	60.5	86.5	89.4	60.2	75.5	93.4	50.6	72.4	88.3
GPT-4-vision	Multimodal	58.5	84.4	86.1	57.5	70.7	90.1	48.5	69.3	84.5
Llava	Multimodal	50.4	71.1	77.5	52.2	67.7	79.2	43.1	66.9	78.1
MuTabNet	Multimodal	57.7	83.2	85.8	56.9	70.1	88.6	47.3	68.6	82.3
Revilio	**Neurosymbolic**	**67.4**	**91.3**	**98.8**	**66.4**	**83.8**	**97.7**	**55.2**	**79.2**	**95.0**

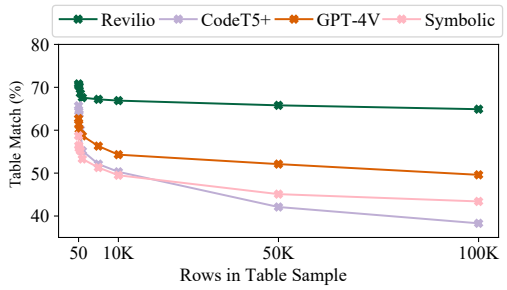


Figure 3: Table match for Revilio and best baselines for each category, across all benchmarks for increasing table sizes. Revilio outperforms all baselines with the performance differential being much more significant in larger tables.

symbolic systems are great at numerical and data summary tables, since these have a clear pattern, but these techniques fail to generalize to semantic tables as they do not have semantic knowledge. Fine-tuned systems like CodeT5+ and CodeLlama suffer from over-specialization, where they split columns and also have inconsistent values in each row. Furthermore, these adapt very poorly to tables with blank values as they tend to hallucinate values. Vision based and multi-modal systems handle empty values and other visual and structural semantics much better due to their additional modality however these struggle to generalize to larger tables.

6.2 Large Tables

A big gap in table parsing systems has been the limitations with the size of tables. Language models are constrained by token limits and vision models are constrained by image sizes. This is further compounded by the cost of larger token counts. Furthermore, even with ever increasing context lengths, it becomes more difficult for these models to maintain performance with increasing table sizes.

Figure 3 shows the top-1 table match performance of Revilio and the best baseline from each category (CodeT5+, GPT4-V and TableLabs) against increasing table sizes. We find that Revilio outperforms baselines across all table sizes, but the performance differential keeps increasing as the table sizes go from 50 rows (+5.1%) to 10K rows (+) and to 100K rows (+25.3%). Further, the average cost for a 10K row table with GPT-4-vision is over 1$ per table, while Revilio is much cheaper and inexpensive due to its short LLM call and symbolic reconstruction engine.

Table 3: Table match for condition learning for different ablations of Revilio. Each ablated component is denoted by '–'. Sketch generation has the highest impact to performance.

System	Wiki	PubTabNet	CleverCSV
Revilio – header detection	62.1%	61.0%	50.9%
Revilio – sketch generation	57.5%	56.2%	46.1%
Revilio – ranker	64.8%	60.2%	53.5%
Revilio – beam search	60.3%	58.5%	49.3%
Revilio	**67.4%**	**66.4%**	**55.0%**

6.3 Design Decisions

We analyze the impact of various components of Revilio. Table 3 shows the top-1 table match over all benchmarks for Revilio and ablated versions created by removing components. We ablate *header detection*, by always treating the first row as the only header row in the table, *sketch generation* by not generating the initial sketch via LLM and directly using the symbolic system without signals from the initial sketch, *ranker* by only using row embedding similarity directly, and *beam search* by performing a greedy search instead.

We find that all components contribute to its performance, with sketch generation having the biggest impact on performance (–10.1%) followed by beam search (–7.8%). This is expected, since the sketch requires semantic knowledge which is provided by the LLM and leveraged by Revilio to extend the sketch to the full table.

7 CONCLUSION

We introduce Revilio, a neuro-symbolic system to reconstruct tables without column boundaries. Revilio detects headers and uses these to generate a table sketch using an LLM. Revilio then does a guided search for the remaining rows to reconstruct the table. We evaluate Revilio on a diverse set of benchmark datasets, demonstrating significant improvements over existing table parsing methods. Furthermore, all the components of Revilio contribute to its overall performance. This work opens up future work in designing neuro-symbolic systems for semantic tabular tasks.

REFERENCES

[1] Eiji Aramaki, Yasuhide Miura, Masatsugu Tonoike, Tomoko Ohkuma, Hiroshi Mashuichi, and Kazuhiko Ohe. 2009. TEXT2TABLE: Medical Text Summarization System Based on Named Entity Recognition and Modality Identification. In *Proceedings of the BioNLP 2009 Workshop*, K. Bretonnel Cohen, Dina Demner-Fushman, Sophia Ananiadou, John Pestian, Jun'ichi Tsujii, and Bonnie Webber (Eds.). Association for Computational Linguistics, Boulder, Colorado, 185–192. https://aclanthology.org/W09-1324

[2] Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2013. Methods for exploring and mining tables on wikipedia. In *Proceedings of the ACM SIGKDD workshop on interactive data exploration and analytics*. 18–26.

[3] Camelot. 2023. PDF Table Extraction for Humans. https://github.com/camelot-dev/camelot. [Online; accessed May-2024].

[4] Wenhu Chen, Xueguang Ma, Xinyi Wang, and William W Cohen. 2023. Program of Thoughts Prompting: Disentangling Computation from Reasoning for Numerical Reasoning Tasks. *Transactions on Machine Learning Research* (2023).

[5] Zewen Chi, Heyan Huang, Heng-Da Xu, Houjin Yu, Wanxuan Yin, and Xian-Ling Mao. 2019. Complicated Table Structure Recognition. arXiv:1908.04729 [cs.IR]

[6] Waleed Farrukh, Antonio Foncubierta-Rodriguez, Anca-Nicoleta Ciubotaru, Guillaume Jaume, Costas Bejas, Orcun Goksel, and Maria Gabrani. 2017. Interpreting Data from Scanned Tables. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Vol. 02. 5–6. https://doi.org/10.1109/ICDAR.2017.250

[7] Wolfgang Gatterbauer, Paul Bohunsky, Marcus Herzog, Bernhard Krüpl, and Bernhard Pollak. 2007. Towards domain-independent information extraction from web tables. In *Proceedings of the 16th international conference on World Wide Web*. 71–80.

[8] Suriya Gunasekar, Yi Zhang, Jyoti Aneja, Caio César Teodoro Mendes, Allie Del Giorno, Sivakanth Gopi, Mojan Javaheripi, Piero Kauffmann, Gustavo de Rosa, Olli Saarikivi, et al. 2023. Textbooks are all you need. *arXiv preprint arXiv:2306.11644* (2023).

[9] Vivek Gupta, Maitrey Mehta, Pegah Nokhiz, and Vivek Srikumar. 2020. INFOTABS: Inference on Tables as Semi-structured Data. arXiv:2005.06117 [cs.CL]

[10] Takaya Kawakatsu. 2024. Multi-Cell Decoder and Mutual Learning for Table Structure and Character Recognition. *arXiv preprint arXiv:2404.13268* (2024).

[11] Thomas Kieninger and Andreas R. Dengel. 1998. The T-Recs Table Recognition and Analysis System. In *International Workshop on Document Analysis Systems*. https://api.semanticscholar.org/CorpusID:38477730

[12] Raymond Li, Yangtian Zi, Niklas Muennighoff, Denis Kocetkov, Chenghao Mou, Marc Marone, Christopher Akiki, LI Jia, Jenny Chim, Qian Liu, et al. 2023. StarCoder: may the source be with you! *Transactions on Machine Learning Research* (2023).

[13] Haotian Liu, Chunyuan Li, Yuheng Li, and Yong Jae Lee. 2023. Improved Baselines with Visual Instruction Tuning.

[14] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. 2023. Visual Instruction Tuning. In *NeurIPS*.

[15] OpenAI. 2024. GPT-4 Technical Report. arXiv:2303.08774 [cs.CL]

[16] Saswat Padhi, Prateek Jain, Daniel Perelman, Oleksandr Polozov, Sumit Gulwani, and Todd Millstein. 2018. FlashProfile: a framework for synthesizing data profiles. *Proceedings of the ACM on Programming Languages* 2, OOPSLA (2018), 1–28.

[17] Shubham Paliwal, Vishwanath D, Rohit Rahul, Monika Sharma, and Lovekesh Vig. 2020. TableNet: Deep Learning model for end-to-end Table detection and Tabular data extraction from Scanned Document Images. arXiv:2001.01469 [cs.CV]

[18] P. Pyreddy and W. B. Croft. 1997. *TINTI: A System for Retrieval in Text Tables TITLE2:*. Technical Report. USA.

[19] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. arXiv:1908.10084 [cs.CL]

[20] Mohammad Mohsin Reza, Syed Saqib Bukhari, Martin Jenckel, and Andreas R. Dengel. 2019. Table Localization and Segmentation using GAN and CNN. *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)* 5 (2019), 152–157. https://api.semanticscholar.org/CorpusID:207950574

[21] Baptiste Rozière, Jonas Gehring, Fabian Gloeckle, Sten Sootla, Itai Gat, Xiaoqing Ellen Tan, Yossi Adi, Jingyu Liu, Romain Sauvestre, Tal Remez, Jérémy Rapin, Artyom Kozhevnikov, Ivan Evtimov, Joanna Bitton, Manish Bhatt, Cristian Canton Ferrer, Aaron Grattafiori, Wenhan Xiong, Alexandre Défossez, Jade Copet, Faisal Azhar, Hugo Touvron, Louis Martin, Nicolas Usunier, Thomas Scialom, and Gabriel Synnaeve. 2024. Code Llama: Open Foundation Models for Code. arXiv:2308.12950 [cs.CL]

[22] Sebastian Schreiber, Stefan Agne, Ivo Wolf, Andreas Dengel, and Sheraz Ahmed. 2017. DeepDeSRT: Deep Learning for Detection and Structure Recognition of Tables in Document Images. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, Vol. 01. 1162–1167. https://doi.org/10.1109/ICDAR.2017.192

[23] Wonkyo Seo, Hyung Il Koo, and Nam Ik Cho. 2014. Junction-based table detection in camera-captured document images. *International Journal on Document Analysis and Recognition (IJDAR)* 18 (2014), 47 – 57. https://api.semanticscholar.org/CorpusID:254106700

[24] Brandon Smock, Rohith Pesala, and Robin Abraham. 2021. PubTables-1M: Towards comprehensive table extraction from unstructured documents. arXiv:2110.00061 [cs.LG]

[25] G. J. J. van den Burg, A. Nazábal, and C. Sutton. 2019. Wrangling Messy CSV Files by Detecting Row and Type Patterns. *Data Mining and Knowledge Discovery* 33, 6 (2019), 1799–1820. https://doi.org/10.1007/s10618-019-00646-y

[26] Jingdong Wang, Ke Sun, Tianheng Cheng, Borui Jiang, Chaorui Deng, Yang Zhao, Dong Liu, Yadong Mu, Mingkui Tan, Xinggang Wang, Wenyu Liu, and Bin Xiao. 2020. Deep High-Resolution Representation Learning for Visual Recognition. arXiv:1908.07919 [cs.CV]

[27] Yue Wang, Hung Le, Akhilesh Deepak Gotmare, Nghi D. Q. Bui, Junnan Li, and Steven C. H. Hoi. 2023. CodeT5+: Open Code Large Language Models for Code Understanding and Generation. arXiv:2305.07922 [cs.CL]

[28] Xueqing Wu, Jiacheng Zhang, and Hang Li. 2022. Text-to-Table: A New Way of Information Extraction. arXiv:2109.02707 [cs.CL]

[29] Zilong Zhao, Robert Birke, and Lydia Chen. 2023. TabuLa: Harnessing Language Models for Tabular Data Synthesis. arXiv:2310.12746 [cs.LG]

[30] Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. 2019. Image-based table recognition: data, model, and evaluation. *arXiv preprint arXiv:1911.10683* (2019).

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009