

Optimized Table Tokenization for Table Structure Recognition

Maksym Lysak^[0000-0002-3723-6960], Ahmed Nassar^[0000-0002-9468-0822],
Nikolaos Livathinos^[0000-0001-8513-3491], Christoph Auer^[0000-0001-5761-0422],
and Peter Staar^[0000-0002-8088-0823]

IBM Research
{mly,ahn,nli,cau,taa}@zurich.ibm.com

Abstract. Extracting tables from documents is a crucial task in any document conversion pipeline. Recently, transformer-based models have demonstrated that table-structure can be recognized with impressive accuracy using Image-to-Markup-Sequence (Im2Seq) approaches. Taking only the image of a table, such models predict a sequence of tokens (e.g. in HTML, LaTeX) which represent the structure of the table. Since the token representation of the table structure has a significant impact on the accuracy and run-time performance of any Im2Seq model, we investigate in this paper how table-structure representation can be optimised. We propose a new, optimised table-structure language (OTSL) with a minimized vocabulary and specific rules. The benefits of OTSL are that it reduces the number of tokens to 5 (HTML needs 28+) and shortens the sequence length to half of HTML on average. Consequently, model accuracy improves significantly, inference time is halved compared to HTML-based models, and the predicted table structures are always syntactically correct. This in turn eliminates most post-processing needs. Popular table structure data-sets will be published in OTSL format to the community.

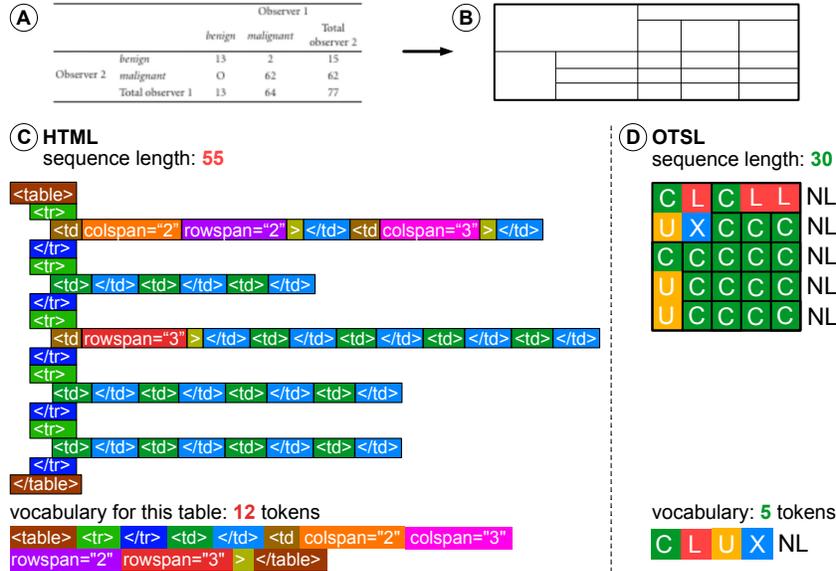
Keywords: Table Structure Recognition · Data Representation · Transformers · Optimization.

1 Introduction

Tables are ubiquitous in documents such as scientific papers, patents, reports, manuals, specification sheets or marketing material. They often encode highly valuable information and therefore need to be extracted with high accuracy. Unfortunately, tables appear in documents in various sizes, styling and structure, making it difficult to recover their correct structure with simple analytical methods. Therefore, accurate table extraction is achieved these days with machine-learning based methods.

In modern document understanding systems [1,15], table extraction is typically a two-step process. Firstly, every table on a page is located with a bounding box, and secondly, their logical row and column structure is recognized. As of

Fig. 1. Comparison between HTML and OTSL table structure representation: (A) table-example with complex row and column headers, including a 2D empty span, (B) minimal graphical representation of table structure using rectangular layout, (C) HTML representation, (D) OTSL representation. This example demonstrates many of the key-features of OTSL, namely its reduced vocabulary size (12 versus 5 in this case), its reduced sequence length (55 versus 30) and an enhanced internal structure (variable token sequence length per row in HTML versus a fixed length of rows in OTSL).



today, *table detection* in documents is a well understood problem, and the latest state-of-the-art (SOTA) object detection methods provide an accuracy comparable to human observers [7,8,10,14,23]. On the other hand, the problem of table structure recognition (TSR) is a lot more challenging and remains a very active area of research, in which many novel machine learning algorithms are being explored [3,4,5,9,11,12,13,14,17,18,21,22].

Recently emerging SOTA methods for table structure recognition employ transformer-based models, in which an image of the table is provided to the network in order to predict the structure of the table as a sequence of tokens. These image-to-sequence (Im2Seq) models are extremely powerful, since they allow for a purely data-driven solution. The tokens of the sequence typically belong to a markup language such as HTML, Latex or Markdown, which allow to describe table structure as rows, columns and spanning cells in various configurations. In Figure 1, we illustrate how HTML is used to represent the table-structure of a particular example table. Public table-structure data sets such as PubTabNet [22], and FinTabNet [21], which were created in a semi-automated way from paired PDF and HTML sources (e.g. PubMed Central), popularized primarily the use of HTML as ground-truth representation format for TSR.

While the majority of research in TSR is currently focused on the development and application of novel neural model architectures, the table structure representation language (e.g. HTML in PubTabNet and FinTabNet) is usually adopted *as is* for the sequence tokenization in Im2Seq models. In this paper, we aim for the opposite and investigate the impact of the table structure representation language with an otherwise unmodified Im2Seq transformer-based architecture. Since the current state-of-the-art Im2Seq model is TableFormer [9], we select this model to perform our experiments.

The main contribution of this paper is the introduction of a new optimised table structure language (OTSL), specifically designed to describe table-structure in an compact and structured way for Im2Seq models. OTSL has a number of key features, which make it very attractive to use in Im2Seq models. Specifically, compared to other languages such as HTML, OTSL has a minimized vocabulary which yields short sequence length, strong inherent structure (e.g. strict rectangular layout) and a strict syntax with rules that only look backwards. The latter allows for syntax validation during inference and ensures a syntactically correct table-structure. These OTSL features are illustrated in Figure 1, in comparison to HTML.

The paper is structured as follows. In section 2, we give an overview of the latest developments in table-structure reconstruction. In section 3 we review the current HTML table encoding (popularised by PubTabNet and FinTabNet) and discuss its flaws. Subsequently, we introduce OTSL in section 4, which includes the language definition, syntax rules and error-correction procedures. In section 5, we apply OTSL on the TableFormer architecture, compare it to TableFormer models trained on HTML and ultimately demonstrate the advantages of using OTSL. Finally, in section 6 we conclude our work and outline next potential steps.

2 Related Work

Approaches to formalize the logical structure and layout of tables in electronic documents date back more than two decades [16]. In the recent past, a wide variety of computer vision methods have been explored to tackle the problem of table structure recognition, i.e. the correct identification of columns, rows and spanning cells in a given table. Broadly speaking, the current deep-learning based approaches fall into three categories: object detection (OD) methods, Graph-Neural-Network (GNN) methods and Image-to-Markup-Sequence (Im2Seq) methods. Object-detection based methods [11,12,13,14,21] rely on table-structure annotation using (overlapping) bounding boxes for training, and produce bounding-box predictions to define table cells, rows, and columns on a table image. Graph Neural Network (GNN) based methods [3,6,17,18], as the name suggests, represent tables as graph structures. The graph nodes represent the content of each table cell, an embedding vector from the table image, or geometric coordinates of the table cell. The edges of the graph define the relationship between the nodes, e.g. if they belong to the same column, row, or table cell.

Other work [20] aims at predicting a grid for each table and deciding which cells must be merged using an attention network. Im2Seq methods cast the problem as a sequence generation task [4,5,9,22], and therefore need an internal table-structure representation language, which is often implemented with standard markup languages (e.g. HTML, LaTeX, Markdown). In theory, Im2Seq methods have a natural advantage over the OD and GNN methods by virtue of directly predicting the table-structure. As such, no post-processing or rules are needed in order to obtain the table-structure, which is necessary with OD and GNN approaches. In practice, this is not entirely true, because a predicted sequence of table-structure markup does not necessarily have to be syntactically correct. Hence, depending on the quality of the predicted sequence, some post-processing needs to be performed to ensure a syntactically valid (let alone correct) sequence.

Within the Im2Seq method, we find several popular models, namely the encoder-dual-decoder model (EDD) [22], TableFormer [9], Tabsplitter[2] and Ye et. al. [19]. EDD uses two consecutive long short-term memory (LSTM) decoders to predict a table in HTML representation. The *tag decoder* predicts a sequence of HTML tags. For each decoded table cell (`<td>`), the attention is passed to the *cell decoder* to predict the content with an embedded OCR approach. The latter makes it susceptible to transcription errors in the cell content of the table. TableFormer address this reliance on OCR and uses two transformer decoders for HTML structure and cell bounding box prediction in an end-to-end architecture. The predicted cell bounding box is then used to extract text tokens from an originating (digital) PDF page, circumventing any need for OCR. TabSplitter [2] proposes a compact double-matrix representation of table rows and columns to do error detection and error correction of HTML structure sequences based on predictions from [19]. This compact double-matrix representation can not be used directly by the Im2seq model training, so the model uses HTML as an intermediate form. Chi et. al. [4] introduce a data set and a baseline method using bidirectional LSTMs to predict LaTeX code. Kayal [5] introduces Gated ResNet transformers to predict LaTeX code, and a separate OCR module to extract content.

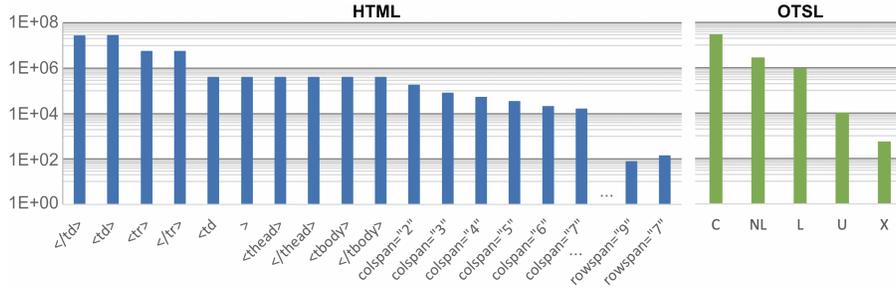
Im2Seq approaches have shown to be well-suited for the TSR task and allow a full end-to-end network design that can output the final table structure without pre- or post-processing logic. Furthermore, Im2Seq models have demonstrated to deliver state-of-the-art prediction accuracy [9]. This motivated the authors to investigate if the performance (both in accuracy and inference time) can be further improved by optimising the table structure representation language. We believe this is a necessary step before further improving neural network architectures for this task.

3 Problem Statement

All known Im2Seq based models for TSR fundamentally work in similar ways. Given an image of a table, the Im2Seq model predicts the structure of the table by generating a sequence of tokens. These tokens originate from a finite vocab-

ulary and can be interpreted as a table structure. For example, with the HTML tokens `<table>`, `</table>`, `<tr>`, `</tr>`, `<td>` and `</td>`, one can construct simple table structures without any spanning cells. In reality though, one needs at least 28 HTML tokens to describe the most common complex tables observed in real-world documents [21,22], due to a variety of spanning cells definitions in the HTML token vocabulary.

Fig. 2. Frequency of tokens in HTML and OTSL as they appear in PubTabNet.



Obviously, HTML and other general-purpose markup languages were not designed for Im2Seq models. As such, they have some serious drawbacks. First, the token vocabulary needs to be artificially large in order to describe all plausible tabular structures. Since most Im2Seq models use an autoregressive approach, they generate the sequence token by token. Therefore, to reduce inference time, a shorter sequence length is critical. Every table-cell is represented by at least two tokens (`<td>` and `</td>`). Furthermore, when tokenizing the HTML structure, one needs to explicitly enumerate possible column-spans and row-spans as words. In practice, this ends up requiring 28 different HTML tokens (when including column- and row-spans up to 10 cells) just to describe every table in the PubTabNet dataset. Clearly, not every token is equally represented, as is depicted in Figure 2. This skewed distribution of tokens in combination with variable token row-length makes it challenging for models to learn the HTML structure.

Additionally, it would be desirable if the representation would easily allow an early detection of invalid sequences on-the-go, before the prediction of the entire table structure is completed. HTML is not well-suited for this purpose as the verification of incomplete sequences is non-trivial or even impossible.

In a valid HTML table, the token sequence must describe a 2D grid of table cells, serialised in row-major ordering, where each row and each column have the same length (while considering row- and column-spans). Furthermore, every opening tag in HTML needs to be matched by a closing tag in a correct hierarchical manner. Since the number of tokens for each table row and column can vary significantly, especially for large tables with many row- and column-spans, it is complex to verify the consistency of predicted structures during sequence

generation. Implicitly, this also means that Im2Seq models need to learn these complex syntax rules, simply to deliver valid output.

In practice, we observe two major issues with prediction quality when training Im2Seq models on HTML table structure generation from images. On the one hand, we find that on large tables, the visual attention of the model often starts to drift and is not accurately moving forward cell by cell anymore. This manifests itself in either in an increasing *location drift* for proposed table-cells in later rows on the same column or even complete loss of vertical alignment, as illustrated in Figure 5. Addressing this with post-processing is partially possible, but clearly undesired. On the other hand, we find many instances of predictions with structural inconsistencies or plain invalid HTML output, as shown in Figure 6, which are nearly impossible to properly correct. Both problems seriously impact the TSR model performance, since they reflect not only in the task of pure structure recognition but also in the equally crucial recognition or matching of table cell content.

4 Optimised Table Structure Language

To mitigate the issues with HTML in Im2Seq-based TSR models laid out before, we propose here our Optimised Table Structure Language (OTSL). OTSL is designed to express table structure with a minimized vocabulary and a simple set of rules, which are both significantly reduced compared to HTML. At the same time, OTSL enables easy error detection and correction during sequence generation. We further demonstrate how the compact structure representation and minimized sequence length improves prediction accuracy and inference time in the TableFormer architecture.

4.1 Language Definition

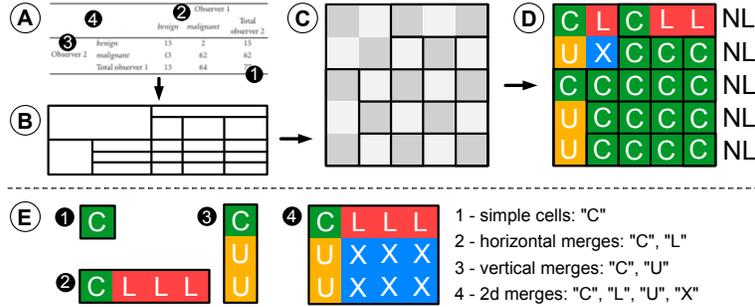
In Figure 3, we illustrate how the OTSL is defined. In essence, the OTSL defines only 5 tokens that directly describe a tabular structure based on an atomic 2D grid.

The OTSL vocabulary is comprised of the following tokens:

- "C" cell - *a new table cell* that either has or does not have cell content
- "L" cell - *left-looking cell*, merging with the left neighbor cell to create a span
- "U" cell - *up-looking cell*, merging with the upper neighbor cell to create a span
- "X" cell - *cross cell*, to merge with both left and upper neighbor cells
- "NL" - *new-line*, switch to the next row.

A notable attribute of OTSL is that it has the capability of achieving lossless conversion to HTML.

Fig. 3. OTSL description of table structure: A - table example; B - graphical representation of table structure; C - mapping structure on a grid; D - OTSL structure encoding; E - explanation on cell encoding



4.2 Language Syntax

The OTSL representation follows these syntax rules:

- Left-looking cell rule:** The left neighbour of an "L" cell must be either another "L" cell or a "C" cell.
- Up-looking cell rule:** The upper neighbour of a "U" cell must be either another "U" cell or a "C" cell.
- Cross cell rule:** The left neighbour of an "X" cell must be either another "X" cell or a "U" cell, and the upper neighbour of an "X" cell must be either another "X" cell or an "L" cell.
- First row rule:** Only "L" cells and "C" cells are allowed in the first row.
- First column rule:** Only "U" cells and "C" cells are allowed in the first column.
- Rectangular rule:** The table representation is always rectangular - all rows must have an equal number of tokens, terminated with "NL" token.

The application of these rules gives OTSL a set of unique properties. First of all, the OTSL enforces a strictly rectangular structure representation, where every new-line token starts a new row. As a consequence, all rows and all columns have exactly the same number of tokens, irrespective of cell spans. Secondly, the OTSL representation is unambiguous: Every table structure is represented in one way. In this representation every table cell corresponds to a "C"-cell token, which in case of spans is always located in the top-left corner of the table cell definition. Third, OTSL syntax rules are only backward-looking. As a consequence, every predicted token can be validated straight during sequence generation by looking at the previously predicted sequence. As such, OTSL can guarantee that every predicted sequence is syntactically valid.

These characteristics can be easily learned by sequence generator networks, as we demonstrate further below. We find strong indications that this pattern

reduces significantly the column drift seen in the HTML based models (see Figure 5).

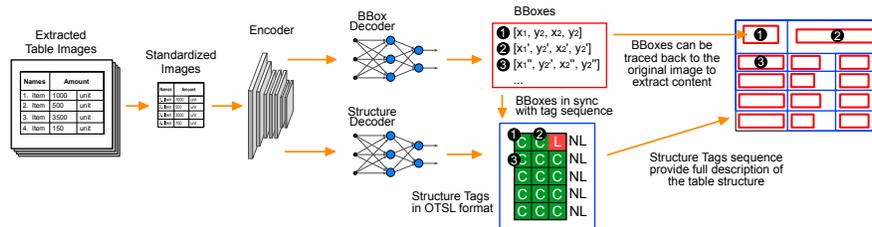
4.3 Error-detection and -mitigation

The design of OTSL allows to validate a table structure easily on an unfinished sequence. The detection of an invalid sequence token is a clear indication of a prediction mistake, however a valid sequence by itself does not guarantee prediction correctness. Different heuristics can be used to correct token errors in an invalid sequence and thus increase the chances for accurate predictions. Such heuristics can be applied either after the prediction of each token, or at the end on the entire predicted sequence. For example a simple heuristic which can correct the predicted OTSL sequence on-the-fly is to verify if the token with the highest prediction confidence invalidates the predicted sequence, and replace it by the token with the next highest confidence until OTSL rules are satisfied.

5 Experiments

To evaluate the impact of OTSL on prediction accuracy and inference times, we conducted a series of experiments based on the TableFormer model (Figure 4) with two objectives: Firstly we evaluate the prediction quality and performance of OTSL vs. HTML after performing Hyper Parameter Optimization (HPO) on the *canonical* PubTabNet data set. Secondly we pick the best hyper-parameters found in the first step and evaluate how OTSL impacts the performance of TableFormer after training on other publicly available data sets (FinTabNet, PubTables-1M [14]). The ground truth (GT) from all data sets has been converted into OTSL format for this purpose, and will be made publicly available.

Fig. 4. Architecture sketch of the TableFormer model, which is a representative for the Im2Seq approach.



We rely on standard metrics such as Tree Edit Distance score (TEDs) for table structure prediction, and Mean Average Precision (mAP) with 0.75 Intersection Over Union (IOU) threshold for the bounding-box predictions of table cells. The predicted OTSL structures were converted back to HTML format in

order to compute the TED score. Inference timing results for all experiments were obtained from the same machine on a single core with AMD EPYC 7763 CPU @2.45 GHz.

5.1 Hyper Parameter Optimization

We have chosen the PubTabNet data set to perform HPO, since it includes a highly diverse set of tables. Also we report TED scores separately for simple and complex tables (tables with cell spans). Results are presented in Table. 1. It is evident that with OTSL, our model achieves the same TED score and slightly better mAP scores in comparison to HTML. However OTSL yields a *2x speed up* in the inference runtime over HTML.

Table 1. HPO performed in OTSL and HTML representation on the same transformer-based TableFormer [9] architecture, trained only on PubTabNet [22]. Effects of reducing the # of layers in encoder and decoder stages of the model show that smaller models trained on OTSL perform better, especially in recognizing complex table structures, and maintain a much higher mAP score than the HTML counterpart.

# enc-layers	# dec-layers	Language	TEDs			mAP (0.75)	Inference time (secs)
			simple	complex	all		
6	6	OTSL	0.965	0.934	0.955	0.88	2.73
		HTML	0.969	0.927	0.955	0.857	5.39
4	4	OTSL	0.938	0.904	0.927	0.853	1.97
		HTML	0.952	0.909	0.938	0.843	3.77
2	4	OTSL	0.923	0.897	0.915	0.859	1.91
		HTML	0.945	0.901	0.931	0.834	3.81
4	2	OTSL	0.952	0.92	0.942	0.857	1.22
		HTML	0.944	0.903	0.931	0.824	2

5.2 Quantitative Results

We picked the model parameter configuration that produced the best prediction quality (enc=6, dec=6, heads=8) with PubTabNet alone, then independently trained and evaluated it on three publicly available data sets: PubTabNet (395k samples), FinTabNet (113k samples) and PubTables-1M (about 1M samples). Performance results are presented in Table. 2. It is clearly evident that the model trained on OTSL outperforms HTML across the board, keeping high TEDs and mAP scores even on difficult financial tables (FinTabNet) that contain sparse and large tables.

Additionally, the results show that OTSL has an advantage over HTML when applied on a bigger data set like PubTables-1M and achieves significantly improved scores. Finally, OTSL achieves faster inference due to fewer decoding steps which is a result of the reduced sequence representation.

Table 2. TSR and cell detection results compared between OTSL and HTML on the PubTabNet [22], FinTabNet [21] and PubTables-1M [14] data sets using TableFormer [9] (with enc=6, dec=6, heads=8).

Data set	Language	TEDs			mAP(0.75)	Inference time (secs)
		simple	complex	all		
PubTabNet	OTSL	0.965	0.934	0.955	0.88	2.73
	HTML	0.969	0.927	0.955	0.857	5.39
FinTabNet	OTSL	0.955	0.961	0.959	0.862	1.85
	HTML	0.917	0.922	0.92	0.722	3.26
PubTables-1M	OTSL	0.987	0.964	0.977	0.896	1.79
	HTML	0.983	0.944	0.966	0.889	3.26

5.3 Qualitative Results

To illustrate the qualitative differences between OTSL and HTML, Figure 5 demonstrates less overlap and more accurate bounding boxes with OTSL. In Figure 6, OTSL proves to be more effective in handling tables with longer token sequences, resulting in even more precise structure prediction and bounding boxes.

Fig. 5. The OTSL model produces more accurate bounding boxes with less overlap (E) than the HTML model (D), when predicting the structure of a sparse table (A), at twice the inference speed because of shorter sequence length (B),(C). "PMC2807444_006_00.png" PubTabNet.

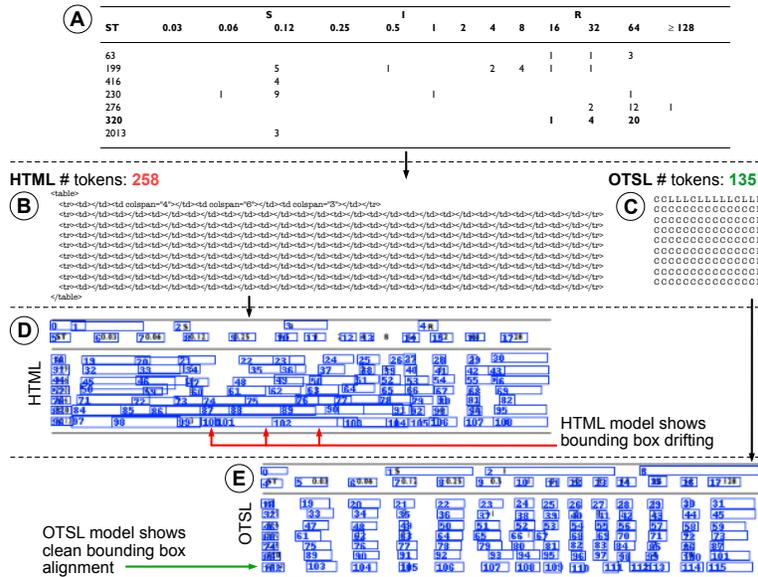
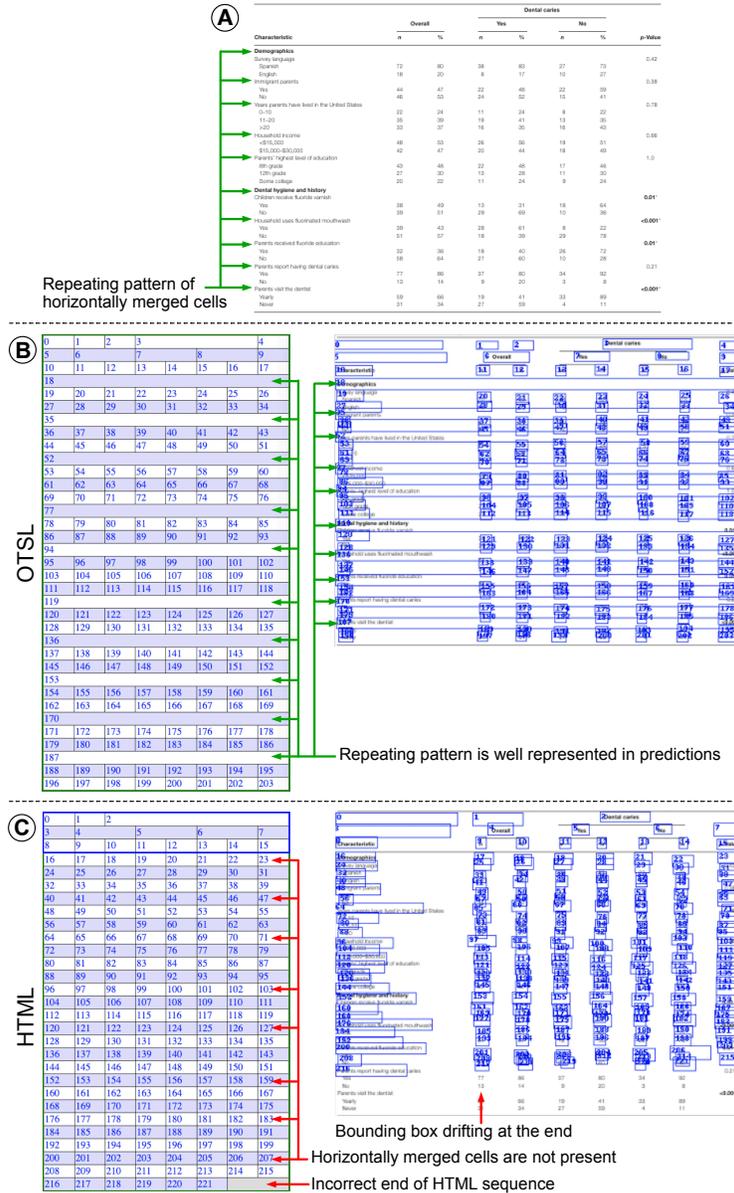


Fig. 6. Visualization of predicted structure and detected bounding boxes on a complex table with many rows. The OTSL model (B) captured repeating pattern of horizontally merged cells from the GT (A), unlike the HTML model (C). The HTML model also didn't complete the HTML sequence correctly and displayed a lot more of drift and overlap of bounding boxes. "PMC5406406_003_01.png" PubTabNet.



6 Conclusion

We demonstrated that representing tables in HTML for the task of table structure recognition with Im2Seq models is ill-suited and has serious limitations. Furthermore, we presented in this paper an Optimized Table Structure Language (OTSL) which, when compared to commonly used general purpose languages, has several key benefits.

First and foremost, given the same network configuration, inference time for a table-structure prediction is about 2 times faster compared to the conventional HTML approach. This is primarily owed to the shorter sequence length of the OTSL representation. Additional performance benefits can be obtained with HPO (hyper parameter optimization). As we demonstrate in our experiments, models trained on OTSL can be significantly smaller, e.g. by reducing the number of encoder and decoder layers, while preserving comparatively good prediction quality. This can further improve inference performance, yielding 5-6 times faster inference speed in OTSL with prediction quality comparable to models trained on HTML (see Table 1).

Secondly, OTSL has more inherent structure and a significantly restricted vocabulary size. This allows autoregressive models to perform better in the TED metric, but especially with regards to prediction accuracy of the table-cell bounding boxes (see Table 2). As shown in Figure 5, we observe that the OTSL drastically reduces the drift for table cell bounding boxes at high row count and in sparse tables. This leads to more accurate predictions and a significant reduction in post-processing complexity, which is an undesired necessity in HTML-based Im2Seq models. Significant novelty lies in OTSL syntactical rules, which are few, simple and always backwards looking. Each new token can be validated only by analyzing the sequence of previous tokens, without requiring the entire sequence to detect mistakes. This in return allows to perform structural error detection and correction on-the-fly during sequence generation.

References

1. Auer, C., Dolfi, M., Carvalho, A., Ramis, C.B., Staar, P.W.J.: Delivering document conversion as a cloud service with high throughput and responsiveness. CoRR **abs/2206.00785** (2022). <https://doi.org/10.48550/arXiv.2206.00785>, <https://doi.org/10.48550/arXiv.2206.00785>
2. Chen, B., Peng, D., Zhang, J., Ren, Y., Jin, L.: Complex table structure recognition in the wild using transformer and identity matrix-based augmentation. In: Porwal, U., Fornés, A., Shafait, F. (eds.) *Frontiers in Handwriting Recognition*. pp. 545–561. Springer International Publishing, Cham (2022)
3. Chi, Z., Huang, H., Xu, H.D., Yu, H., Yin, W., Mao, X.L.: Complicated table structure recognition. arXiv preprint arXiv:1908.04729 (2019)
4. Deng, Y., Rosenberg, D., Mann, G.: Challenges in end-to-end neural scientific table recognition. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. pp. 894–901. IEEE (2019)

5. Kayal, P., Anand, M., Desai, H., Singh, M.: Tables to latex: structure and content extraction from scientific tables. *International Journal on Document Analysis and Recognition (IJDAR)* pp. 1–10 (2022)
6. Lee, E., Kwon, J., Yang, H., Park, J., Lee, S., Koo, H.I., Cho, N.I.: Table structure recognition based on grid shape graph. In: *2022 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*. pp. 1868–1873. IEEE (2022)
7. Li, M., Cui, L., Huang, S., Wei, F., Zhou, M., Li, Z.: Tablebank: A benchmark dataset for table detection and recognition (2019)
8. Livathinos, N., Berrospi, C., Lysak, M., Kuropiatnyk, V., Nassar, A., Carvalho, A., Dolfi, M., Auer, C., Dinkla, K., Staar, P.: Robust pdf document conversion using recurrent neural networks. *Proceedings of the AAAI Conference on Artificial Intelligence* **35**(17), 15137–15145 (May 2021), <https://ojs.aaai.org/index.php/AAAI/article/view/17777>
9. Nassar, A., Livathinos, N., Lysak, M., Staar, P.: Tableformer: Table structure understanding with transformers. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 4614–4623 (June 2022)
10. Pfitzmann, B., Auer, C., Dolfi, M., Nassar, A.S., Staar, P.W.J.: Doclaynet: A large human-annotated dataset for document-layout segmentation. In: Zhang, A., Rangwala, H. (eds.) *KDD '22: The 28th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, August 14 - 18, 2022. pp. 3743–3751. ACM (2022). <https://doi.org/10.1145/3534678.3539043>, <https://doi.org/10.1145/3534678.3539043>
11. Prasad, D., Gadpal, A., Kapadni, K., Visave, M., Sultanpure, K.: 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*. pp. 572–573 (2020)
12. Schreiber, S., Agne, S., Wolf, I., Dengel, A., Ahmed, S.: 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. 1, pp. 1162–1167. IEEE (2017)
13. Siddiqui, S.A., Fateh, I.A., Rizvi, S.T.R., Dengel, A., Ahmed, S.: Deeptabstr: Deep learning based table structure recognition. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. pp. 1403–1409 (2019). <https://doi.org/10.1109/ICDAR.2019.00226>
14. Smock, B., Pesala, R., Abraham, R.: PubTables-1M: Towards comprehensive table extraction from unstructured documents. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. pp. 4634–4642 (June 2022)
15. Staar, P.W.J., Dolfi, M., Auer, C., Bekas, C.: Corpus conversion service: A machine learning platform to ingest documents at scale. In: *Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. pp. 774–782. *KDD '18*, Association for Computing Machinery, New York, NY, USA (2018). <https://doi.org/10.1145/3219819.3219834>, <https://doi.org/10.1145/3219819.3219834>
16. Wang, X.: *Tabular Abstraction, Editing, and Formatting*. Ph.D. thesis, CAN (1996), aAINN09397
17. Xue, W., Li, Q., Tao, D.: Res2tim: Reconstruct syntactic structures from table images. In: *2019 International Conference on Document Analysis and Recognition (ICDAR)*. pp. 749–755. IEEE (2019)

18. Xue, W., Yu, B., Wang, W., Tao, D., Li, Q.: Tgrnet: A table graph reconstruction network for table structure recognition. In: Proceedings of the IEEE/CVF International Conference on Computer Vision. pp. 1295–1304 (2021)
19. Ye, J., Qi, X., He, Y., Chen, Y., Gu, D., Gao, P., Xiao, R.: Pingan-vcgroup’s solution for icdar 2021 competition on scientific literature parsing task b: Table recognition to html (2021). <https://doi.org/10.48550/ARXIV.2105.01848>, <https://arxiv.org/abs/2105.01848>
20. Zhang, Z., Zhang, J., Du, J., Wang, F.: Split, embed and merge: An accurate table structure recognizer. *Pattern Recognition* **126**, 108565 (2022)
21. Zheng, X., Burdick, D., Popa, L., Zhong, X., Wang, N.X.R.: Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context. In: 2021 IEEE Winter Conference on Applications of Computer Vision (WACV). pp. 697–706 (2021). <https://doi.org/10.1109/WACV48630.2021.00074>
22. Zhong, X., ShafieiBavani, E., Jimeno Yepes, A.: Image-based table recognition: Data, model, and evaluation. In: Vedaldi, A., Bischof, H., Brox, T., Frahm, J.M. (eds.) *Computer Vision – ECCV 2020*. pp. 564–580. Springer International Publishing, Cham (2020)
23. Zhong, X., Tang, J., Yepes, A.J.: Publaynet: largest dataset ever for document layout analysis. In: 2019 International Conference on Document Analysis and Recognition (ICDAR). pp. 1015–1022. IEEE (2019)