Paper ML12

From Breaking Bad to Breaking Data: Unlocking TFL PDFs

Andras Kasa, UCB Pharma S.A., Madrid, Spain
Darius Tetsa, UCB Pharma S.A , Slough, United Kingdom

ABSTRACT

Pharmaceutical companies hold vast unstructured data, such as PDF TFL outputs, which are difficult for machines to interpret due to complex layouts with headers, bodies, and footers across multiple hierarchies. While text extraction from PDFs is straightforward, preserving structural organization and semantic meaning remains a challenge. Our work explores layout understanding and content segmentation of standard TFL outputs using techniques like clustering and computer vision. We highlight the potential of combining computer vision with natural language processing to further improve results. By enabling machines to better understand document structure and semantics, this research lays the groundwork for advanced downstream applications such as LLM-supported RAGs, metadata recommendation, process standardization, automated document generation, and SAS® or R program skeleton creation.

INTRODUCTION

In today's technological landscape, leveraging artificial intelligence and machine learning techniques to gain insights from data has become a game changer. Organizations are increasingly adopting these technologies to enhance their data-driven decision-making processes. Significant advancements have been made in tools and techniques for extracting content and insights from unstructured data. These tools have evolved from simple text extraction to sophisticated methods like semantic chunking. This evolution underscores the importance of context in generating high-quality knowledge from unstructured documents. Human readers use various cues such as context, conventions, and language information, along with complex reasoning, to interpret document contents. However, automatically analysing documents with complex layouts remains a challenging task [2]. In clinical trials, TFLs (Tables, Figures, and Listings) are typically provided as PDF documents with standard yet complex layouts, combining both structured and unstructured content. Our goal is to develop a tool that can unlock the structure of TFL outputs, facilitating content findability, reusability and insight generation. In the following sections, we will explore state-of-the-art techniques we have experimented with and their outcomes.

BACKGROUND

UCB's TFL outputs serve as critical documentation in clinical trials, containing essential analysis results and supporting information. While UCB maintains minimal technical requirements, outputs from vendors, internal teams, and those generated in R can vary in where sections are located. UCB's TFL outputs have evolved over time, with older versions differing from recent ones and occasional deviations from the standard layout have been noted. This variability poses challenges for automated processing and content extraction.

Despite these variations, TFL documents typically follow a common structural pattern consisting of four main sections (see Figure 1):

The Header section contains deliverable information associated with the TFL, such as the sponsor, study code, product code, document confidentiality status, and document editing status.

The Title section contains the title of the output.

The Body section contains the actual tables, figures, or listings of interest. It may also include a grouping variable for the analysis results in the TFL document.

The Footer section contains information to help facilitate understanding of the body section. This may include abbreviations or definitions of terms, references to other TFL output documents, the program names used to generate the TFL PDF file, the timestamp of the program run, and paging information.

Note: TFL outputs may be further specialized into sub sections.

Figure 1: TFL document structure showing the four main sections: Header, Title, Body, and Footer.

Unlocking the structure of these documents could automate metadata extraction, aiding in-house reprogramming of tables, listings, or figures. It also supports building knowledge graphs for our data assets, creating a semantic layer to address complex data findability questions and reveal hidden connections in our outputs.

EXPERIMENTS

Our objective was to build a tool capable of recognizing the TFL layout and independently extracting its various sections. We used Python for our experiments.

PDF EXTRACTION LIBRARIES

During the initial phase, we tested several techniques, beginning with a few Python libraries, including EasyOCR, PyPDF2, PyMuPDF, and pdfminer. Table 1 summarizes the performance of these libraries in extracting and interpreting the semantics of our PDF TFL outputs.

Library	Text Extraction	OCR Recognition	TFL Layout Recognition
PyPDF2™	Good	Good	Poor
PyMuPDF™	Good	Excellent	Poor
pdfminer™	Good	Poor	Poor
EasyOCR™	Good	Good	Poor

Table 1: Summary of PDF data extraction tools.

From Table 1, we can see that all the PDF extraction tools did a decent job pulling text from the TFL outputs, but none of them were able to understand the structure of the documents. We also noticed inconsistencies in how the content was read. In some cases, it was processed line by line, while in others it was read from right to left. The tools also lacked the sophistication needed to detect table structures and handle line breaks properly.

One of the main challenges we encountered was with coordinate-based extraction methods. These approaches rely on the assumption that document elements stay in fixed positions. However, in our TFL outputs, we found cases where footer content appeared in the middle of the page, disrupting the layout. Because of this spatial inconsistency, simple position-based rules were not able to accurately separate the different sections of the document.

HIERARCHICAL CLUSTERING

Because the results for layout recognition using PDF data extraction tools were unsatisfactory, we decided to explore alternative options and turned to machine learning. We began with an unsupervised learning approach, specifically hierarchical clustering, as these techniques typically do not require pre-training.

Hierarchical clustering is an unsupervised learning technique that groups similar objects into clusters by creating a hierarchy through merging or splitting based on similarity measures [1, 3]. This method is advantageous because it does not require specifying the number of clusters in advance and can utilize any valid distance measure. However, it can be computationally intensive and is sensitive to the choice of distance metric and linkage criterion. Hierarchical clustering has been widely applied in pattern recognition and data analysis tasks, including document layout analysis and image segmentation.

After applying hierarchical clustering to our TFL outputs, the results were not satisfactory. Although the algorithm consistently isolated the footer into a distinct cluster, it typically combined the header and title into the same cluster, and the remaining clusters were generally indistinct. More critically, the clustering approach still relied on spatial features and could not account for the semantic meaning of content or handle cases where sections appeared in non-standard positions.

These experiments with classical segmentation techniques and coordinate based methods demonstrated a fundamental limitation: without understanding the semantic content and context of document elements, purely geometric or clustering approaches cannot reliably segment TFL documents with variable layouts. This realization motivated our exploration of more sophisticated approaches capable of learning document structure and semantic relationships.

[1] PyPDF2™: https://pypdf2.readthedocs.io/en/3.x/
[2] PyMuPDF™: https://pymupdf.readthedocs.io/
[3] pdfminer™: https://pypi.org/project/pdfminer/
[4] EasyOCR™: https://pypi.org/project/easyocr/1.1.4/

Computer vision and supervised learning have been proven to perform well in object recognition. We believed

that applying them to our PDF outputs could enable us to learn the patterns in our outputs by training a model to recognize those patterns.

METHODOLOGY

Our approach follows four key steps:

1. We first selected a pre-trained machine learning model with proven performance in pattern recognition.

2. We then created an annotated training dataset with clearly labeled sections of interest.

3. Using transfer learning techniques, we fine-tuned our model to identify these specific sections in TFL outputs.

4. Finally, we tested our trained model on new documents to evaluate its ability to accurately identify key elements in TFL documents.

TOOLS AND TECHNOLOGY: YOLO

We trained YOLO 8.0 to recognize the layout of the TFL outputs. YOLO® (You Only Look Once) is a popular object detection and image segmentation model developed by Joseph Redmon and Ali Farhadi at the University of Washington. Launched in 2015, YOLO is a deep learning and computer vision algorithm which quickly gained popularity for its high speed and accuracy [4]. Unlike traditional models that rely on region proposal techniques requiring thousands of iterations, YOLO requires only one forward propagation pass through the neural network to make predictions. This makes it significantly faster and more efficient, ideal for real-time applications.

LABELIMG IMPLEMENTATION

LabelImg was used for the annotation of our outputs. LabelImg is a free, open-source graphical image annotation tool written in Python. It provides a simple and intuitive interface for creating bounding boxes and labeling objects in images, making it particularly accessible due to its straightforward installation process. The tool outputs annotations in formats compatible with popular object detection frameworks, including YOLO, Pascal VOC, and CreateML.

To maintain focus in our initial experiments, we limited our scope to tables and listings, as they share fairly similar structural characteristics. For our implementation, we:

1. Annotated approximately 736 Table and Listing outputs using four distinct labels: Header, Title, Body, and Footer.

2. Selected a diverse collection of Table and Listing outputs to ensure our training set represented the full variety of UCB's document structures.

3. We deliberately selected tables from various vendors and internal sources, each with varying spatial features, to capture a wide range of document layouts and arrangements for our training data.

4. Converted the annotated outputs into image files with YOLO-format annotations.

5. Trained YOLO on these annotated images to recognize the different document sections, using 210 images in the validation set during training.

6. Validated the trained model by testing its section prediction capabilities on 106 independent test files.

ANNOTATION REFINEMENT AND DATA AUGMENTATION

During our initial annotation process, we noticed that some bounding boxes were too tight, meaning they did not fully cover some characters of the text. To address this, we adjusted the annotations to make the bounding boxes wider, ensuring complete text coverage.

Additionally, we applied data augmentation techniques during YOLO training to improve model robustness. Specif-

ically, we used copy_paste=0.1 and mixup=0.2 augmentation strategies. The copy_paste augmentation randomly pastes objects from one image onto another, helping the model learn to recognize document sections in varying contexts and spatial arrangements. The mixup augmentation blends two training images and their labels, creating synthetic training examples that encourage the model to learn more generalized features rather than memorizing specific layouts. These techniques are particularly valuable for our use case, as TFL documents can have significant layout variability, and augmentation helps the model generalize better to unseen document structures without requiring additional manual annotations.

TRAINING RESULTS

The model was trained for 50 epochs with impressive results (see Figure 2). Table 2 summarizes the key performance metrics achieved:

Metric	Value
Precision	0.991
Recall	0.996
mAP@50	0.994
mAP@50-95	0.899

Table 2: Final YOLO model performance metrics on validation set.

The model demonstrated excellent performance across all four document sections. All classes achieved mAP@0.5 scores above 0.99, with the overall model reaching 0.994. The model maintains high F1 scores (0.99) at a confidence threshold of 0.698, indicating reliable predictions across all section types.

UNDERSTANDING THE EVALUATION METRICS

To assess how well our model performed, we used several widely accepted metrics from the field of object detection:

Precision tells us how many of the sections detected by the model were actually correct. With a precision score of 0.991, it means that 99.1% of the sections identified were classified accurately.

Recall shows how many of the actual sections in the documents were successfully found by the model. A recall of 0.996 indicates that the model detected 99.6% of all existing sections.

mAP@50 (mean Average Precision at an IoU threshold of 0.5) measures how closely the predicted bounding boxes match the true section boundaries. An IoU (Intersection over Union) of 0.5 means the predicted and actual boxes must overlap by at least 50%. Our score of 0.994 reflects excellent accuracy in locating sections.

mAP@50–95 is a more demanding metric that averages performance across a range of IoU thresholds, from 0.5 up to 0.95. Our score of 0.899 shows that the model maintains strong performance even when precise boundary alignment is required.

F1 Score combines precision and recall into a single value, giving a balanced view of the model's overall effectiveness. The F1-confidence curve helps identify the best confidence threshold for making predictions.

Confusion Matrix visualizes classification performance, showing how often each section type is correctly identified versus misclassified as another type. The diagonal elements represent correct classifications, while off-diagonal elements indicate errors.

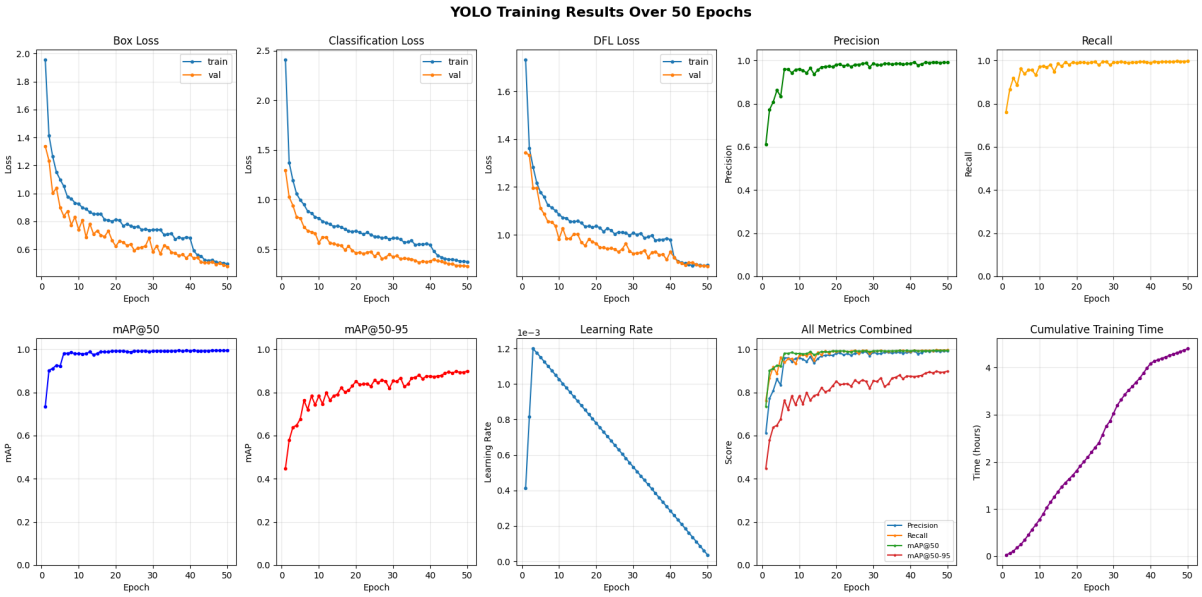

Figure 2: Training and validation metrics across 50 epochs, showing convergence of loss functions and improvement in precision, recall, and mAP scores.

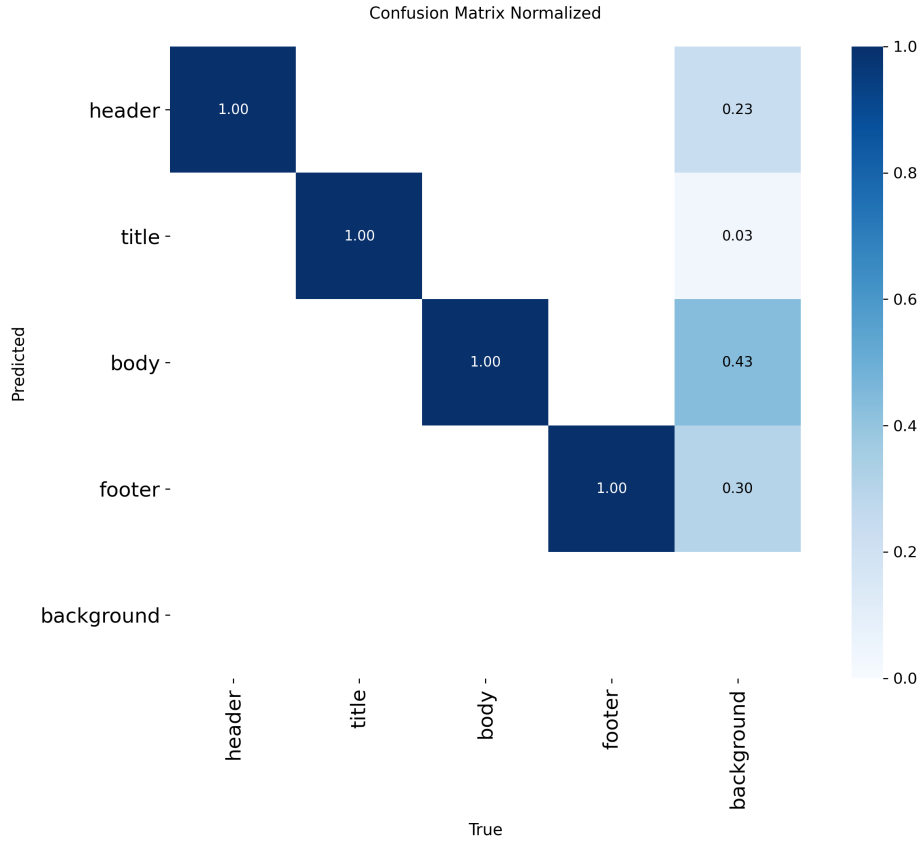

Figure 3: Normalized confusion matrix showing near-perfect classification accuracy across all document sections.

The confusion matrix (Figure 3) reveals minimal misclassifications, with the model correctly identifying 209 headers, 210 titles, 209 body sections, and 210 footers in the validation set. The normalized confusion matrix shows classification accuracies of 100% for headers, titles, and footers, and the body section also achieving 100% accuracy, although with a small number of background misclassifications (0.43).

These results demonstrate that supervised learning with YOLO successfully addresses the limitations of classical segmentation techniques, enabling accurate and reliable identification of the TFL document structure regardless of spatial layout variations.

POST-YOLO BOUNDARY REFINEMENT

While our YOLO model achieves excellent section classification accuracy (100% across all categories), precise boundary positioning between adjacent sections presents challenges. Figure 4 illustrates this issue: the upper boundary of the body bounding box intersects the "By" grouping statement, which should ideally be included within the body region for complete table context.

Figure 4: Raw YOLO bounding boxes showing boundary imprecision. The upper edge of the body bounding box (red) intersects the "By" grouping statement, which should be included in the body region for complete table extraction.

Such boundary imprecision can lead to incomplete text extraction when cropping sections based on YOLO predictions. We address this through a two-phase refinement strategy.

Phase 1: Geometric Refinement. For each pair of adjacent sections (S1, S2), we identify the optimal separating line between their YOLO-predicted bounding boxes. Given the lower boundary of the upper box ($S1_{y_{lower}}$) and the

upper boundary of the lower box ($S2_{y_{upper}}$), we calculate the separation line as:

$$y_{S1,S2} = \frac{S1_{y_{lower}} + S2_{y_{upper}}}{2} \tag{1}$$

This approach, inspired by support vector machine theory [5], finds the Y-coordinate that maximally separates the two sections. We also implement a weighted variant that accounts for YOLO confidence scores when calculating this average. Figure 5 demonstrates the refined boundaries, where the separating lines are positioned in the whitespace between sections, ensuring clean separation without content truncation.

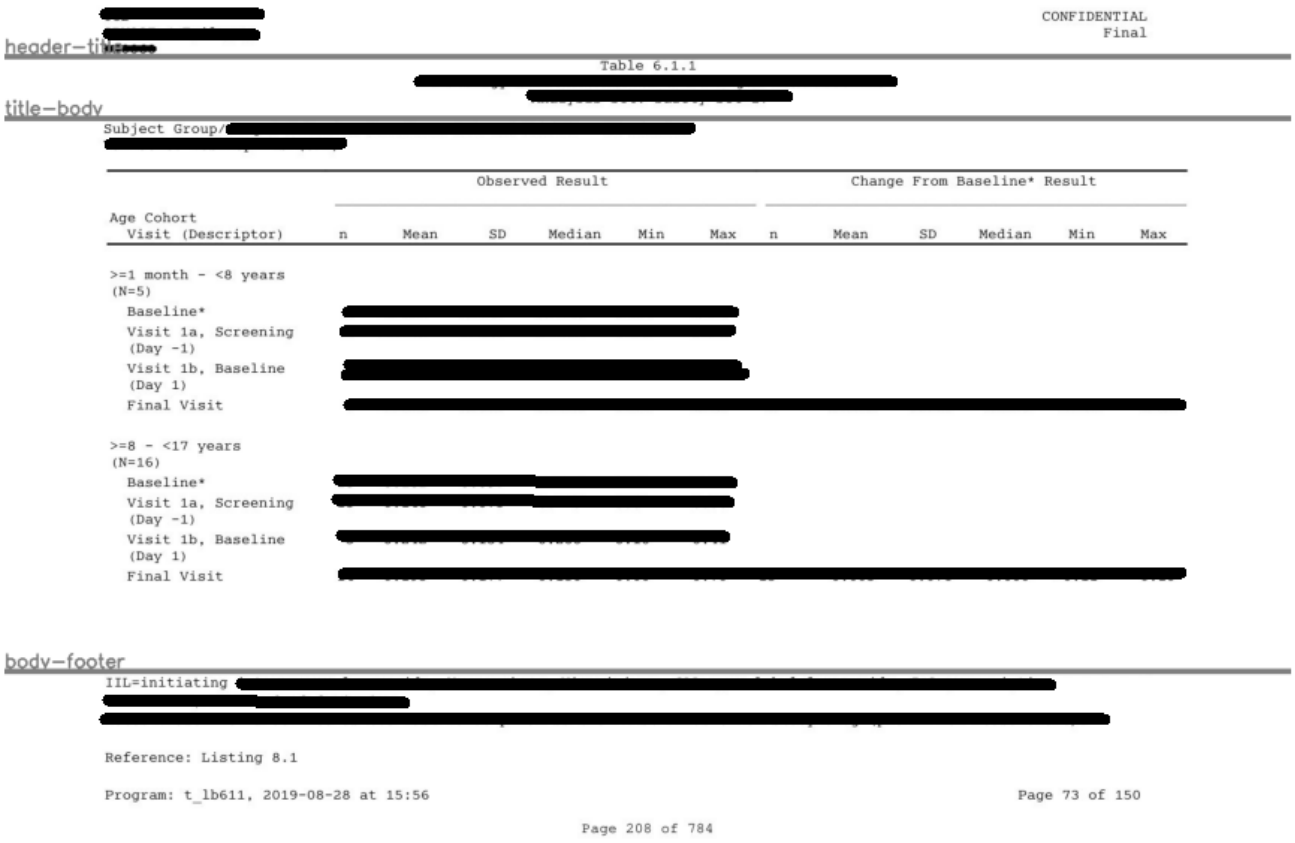

Figure 5: Refined boundaries using geometric averaging. The gray lines represent the calculated separation boundaries positioned in whitespace between sections, resolving the text truncation issue shown in Figure 4.

Phase 2: Consensus Voting. To establish output-wide boundaries efficiently, we employ strategic sampling designed to detect layout variations:

- Output $\geq$ 12 pages: Sample first 5 pages + 6 randomly selected pages + last page ($\sim$12 pages)
- Output $<$ 12 pages: Process all pages

The first pages are sampled because they rarely show layout anomalies: first pages may contain footer content positioned in the body region, or some listings employ alternating column layouts where odd and even pages display different column sets, requiring early-page sampling to detect this pattern. The last page is sampled because final pages frequently contain incomplete content (single-line body sections, truncated footers), representing edge cases that could skew boundary detection if not identified.

For each section boundary (header-title, title-body, body-footer), we aggregate refined positions from sampled pages using the *mean* (or optionally weighted mean based on confidence scores). Prior to aggregation, we normalize text content by removing page-specific patterns (e.g., "Page 1 of 53", "Page 2 of 53") from headers and footers, as these unique identifiers would otherwise cause each page to be treated as a distinct layout variant, disrupting the consensus voting mechanism. Pages identified as anomalous (e.g., footer content in body region) are excluded from voting. The consensus boundaries are then applied uniformly across all document pages.

Our experiments demonstrated near-perfect agreement (approaching 100% consensus) across sampled pages, confirming that boundary positions remain highly consistent throughout documents and validating our geometric refinement approach.

This approach significantly reduces YOLO inference costs while capturing layout variations that pure random sampling might miss.

TABLE STRUCTURE PRESERVATION WITH PYMUPDF

Following boundary detection, we extract structured content from body regions using PyMuPDF's layout-aware text extraction. Unlike sequential text extraction methods, PyMuPDF's dictionary-based approach (`get_text("dict")`) preserves spatial relationships between text elements, maintaining the original tabular layout through whitespace positioning and line breaks.

Our extraction pipeline groups text elements by vertical position (Y-coordinate) to identify rows, then sorts horizontally (X-coordinate) within each row to maintain column order. The resulting output preserves the spatial arrangement of text through appropriate spacing, combined with line breaks to separate rows. This preserves the table structure using spaces and line breaks, maintaining column alignment as in the original PDF.

This structure-preserving format proves particularly valuable for integration with large language models (LLMs). The preserved layout, with spatial alignment and explicit row breaks, enables LLMs to accurately interpret tabular relationships and content structure. Figure 6 demonstrates this capability: the raw PyMuPDF extraction maintains column alignment through whitespace, which modern LLMs (such as Microsoft Copilot®) successfully interpret to reconstruct formatted tables and answer complex queries about the statistical content without requiring additional parsing logic.

[redacted]: <=2 years of age (N=21)

Baseline Health State	Year 1 Health State: Early Ambulatory n (%)	Late Ambulatory n (%)	Non Ambulatory n (%)	Ambulatory NIV 4+ n (%)	Non Ambulatory NIV 4+ n (%)	Non Ambulatory IV 8+ n (%)	Death n (%)
Early Ambulatory	[redacted]						
Late Ambulatory	[redacted]						
Non Ambulatory	[redacted]						
Ambulatory NIV 4+	[redacted]						
Non Ambulatory NIV 4+	[redacted]						
Non Ambulatory IV 8+	[redacted]						
Death	[redacted]						

[redacted]: ≤ 2 years of age (N = 21)

Year 1 Health State

Baseline Health State	Early Ambulatory	Late Ambulatory	Non Ambulatory	Ambulatory NIV 4+	Non Ambulatory NIV 4+	Non Ambulatory IV 8+	Death
Early Ambulatory	[redacted]						
Late Ambulatory	[redacted]						
Non Ambulatory	[redacted]						
Ambulatory NIV 4+	[redacted]						
Non Ambulatory NIV 4+	[redacted]						
Non Ambulatory IV 8+	[redacted]						
Death	[redacted]						

Figure 6: Demonstration of structure-preserving extraction and LLM interpretation. PyMuPDF maintains table layout through spatial positioning and line breaks, which enables modern language models (Microsoft Copilot shown here) to accurately reconstruct the original table structure. Note: numerical values have been redacted for confidentiality; validation confirmed accurate structure recognition and content interpretation.

This LLM-compatible output format supports downstream applications including automated question-answering systems, metadata extraction for knowledge graphs, and content verification workflows, bridging the gap between PDF documents and modern language model capabilities.

DISCUSSION

The results of this experiment highlight the potential of combining computer vision and supervised learning for structural segmentation of complex TFL (Tables, Figures, Listings) outputs. The YOLO-based model demonstrated excellent classification accuracy across all document sections, outperforming traditional heuristic-based and unsupervised methods. This success underscores the importance of layout-aware modeling in domains where document structure is both standardized yet exhibits vendor-specific variations.

THE SAVE FRAMEWORK: PRACTICAL APPLICATIONS

Our structural segmentation pipeline enables four key application areas, which we term the SAVE framework:

Searchable Knowledge Base (S): By extracting structured sections from historical TFL outputs, we can build a comprehensive repository of past studies, methodologies, and statistical implementations. This enables programmers to search for specific analytical approaches (e.g., "all studies using multiple imputation on questionnaire X") and retrieve relevant examples with their associated Statistical Analysis Plans and implementation details. Such searchability transforms institutional knowledge from isolated documents into an accessible, queryable resource.

Automated Reasoning (A): The structured metadata extracted from body sections, combined with header and footer

information, provides context for AI-based reasoning systems. Large language models can leverage this structured content to recommend analytical approaches, identify similar past analyses, and support decision-making. As demonstrated in Figure 6, modern LLMs successfully interpret our extracted table structures, enabling automated question-answering about statistical results and methodological choices. This capability addresses known limitations of LLMs in clinical programming by grounding their reasoning in validated historical patterns.

Validation & Quality Control (V): Automated extraction of statistical results from body sections enables systematic validation of vendor-delivered outputs. By extracting population counts (big N), endpoint calculations, and consistency metrics across table columns, quality control systems can verify correctness without manual inspection. This automated validation reduces the risk of accepting erroneous vendor deliverables while accelerating the review process, a critical capability for maintaining data integrity in regulatory submissions.

Extractable Metadata (E): Perhaps the most immediate practical benefit lies in metadata generation for vendor-outsourced analyses. When vendors provide TFL outputs but limited documentation, our pipeline automatically extracts titles, footnotes, population definitions, and analytical parameters from the structured sections. This extracted metadata enables rapid in-house reprogramming for ad-hoc subgroup analyses or sensitivity analyses, reducing vendor dependency and improving programming agility. The metadata also facilitates program skeleton generation, where extracted structural information guides the creation of new analytical programs adapted to the internal computing environment.

BROADER IMPLICATIONS AND REMAINING CHALLENGES

These capabilities are particularly valuable in pharmaceutical environments where consistency, traceability, and regulatory compliance are paramount. The pipeline's applicability extends beyond individual document processing to support large-scale knowledge management initiatives, including the construction of knowledge graphs that link studies, methodologies, outcomes, and analytical decisions across the clinical development portfolio.

However, several challenges remain. The reliance on annotated training data (736 documents in our case) introduces scalability concerns when expanding to new document types or therapeutic areas. While the model generalizes well across UCB's TFL layouts and multiple vendor formats, further validation is needed to assess performance on legacy systems with non-standard formatting or outputs.

The boundary refinement and consensus voting approach handles layout variability well, though occasional irregular documents may require additional exception handling beyond our current anomaly detection.

FUTURE DIRECTIONS

Future work should explore tighter integration of natural language processing (NLP) techniques beyond the current LLM-based interpretation demonstrated in our validation. Specifically, combining layout segmentation with named entity recognition could enable extraction of protocol numbers, study phases, treatment arms, and endpoint definitions directly from unstructured text within sections. This semantic layer would enrich the metadata and support more sophisticated reasoning capabilities.

Additionally, extending the pipeline to handle figures would broaden applicability. While our current approach successfully processes both tables and listings (which share similar structural characteristics), figures often contain visual encodings of statistical relationships that require different extraction techniques, potentially leveraging multimodal vision-language models.

Finally, the human-centric design of the pipeline emphasizes usability, interpretability, and integration with existing workflows, positioning it as a practical tool for biostatisticians, statistical programmers, and data scientists. The near-perfect boundary consensus (approaching 100% agreement) across sampled pages suggests that the approach captures genuine structural regularities in TFL documents rather than overfitting to specific training examples. Continued collaboration between technical teams and end-users will be essential to refine the solution, expand its cover-

age, and ensure alignment with evolving regulatory requirements and business needs.

CONCLUSION

This work demonstrates the feasibility and effectiveness of applying computer vision and supervised learning techniques to the structural segmentation of complex TFL PDF outputs. While traditional PDF extraction tools and unsupervised clustering methods provided limited success due to their reliance on spatial heuristics, our YOLO-based approach achieved near-perfect classification accuracy across all document sections: Header, Title, Body, and Footer.

Beyond classification, our geometric boundary refinement and consensus voting strategy addresses the critical challenge of precise section separation. The approach achieves near-perfect boundary agreement (approaching 100% consensus) across sampled pages while reducing computational costs by 85% through strategic sampling. Combined with PyMuPDF's structure-preserving extraction, the complete pipeline produces LLM-compatible output that enables downstream automation, validation, and knowledge management applications embodied in our SAVE framework: Searchable knowledge bases, Automated reasoning, Validation systems, and Extractable metadata for vendor independence.

By leveraging annotated outputs, transfer learning, and data augmentation strategies, we successfully trained a robust model capable of generalizing across diverse TFL layouts. These results underscore the potential of combining layout-aware computer vision with semantic understanding to unlock unstructured clinical trial output metadata extraction.

Looking ahead, this foundational work opens the door to a broader pipeline of downstream applications, including automated metadata generation, knowledge graph construction, and programmatic content reuse. Continued refinement of annotation strategies, expansion to additional document types (particularly figures), and integration with natural language processing models will further enhance the utility and scalability of this solution across the clinical documentation lifecycle.

REFERENCES

[1] Jain, A. K., & Dubes, R. C. (1988). *Algorithms for clustering data*. Prentice Hall.

[2] Namboodiri, A. M., & Jain, A. K. (2007). Document structure and layout analysis. In *Digital Document Processing* (pp. 29–48). Springer.

[3] Nielsen, F. (2016). Hierarchical Clustering. In *Introduction to HPC with MPI for Data Science. Undergraduate Topics in Computer Science*. Springer, Cham. https://doi.org/10.1007/978-3-319-21903-5_8

[4] Terven, J., Córdova-Esparza, D., and Romero-González, J. A. (2023). A comprehensive review of YOLO architectures in computer vision: From YOLOv1 to YOLOv8 and YOLO-NAS. *Machine Learning and Knowledge Extraction*, 5(4), 1680-1716.

[5] Cortes, C., & Vapnik, V. (1995). Support-vector networks. *Machine Learning*, 20(3), 273–297. Springer. https://doi.org/10.1007/BF00994018

ACKNOWLEDGMENTS

We would like to thank our management team, and especially Alberto Montironi, Mike Branson, Phyllis Semtana, and Tim Williams, for their continuous support. We also extend our gratitude to our colleagues Callum Charalambides, Ethan Pollitt, Kevin Ray, and Richard Abdy for their valuable contributions to this project.

Furthermore, Andras Kasa would like to express his deep appreciation to his father for introducing him to programming at a young age. This early exposure to programming has had a profound impact on his life and professional

journey. Darius Tetsa would like to express his deepest gratitude to his mom, whose unwavering support and guidance have been instrumental in shaping his journey.

CONTACT INFORMATION

Your comments and questions are valued and encouraged. Contact the author at:

Author Name: Andras Kasa
Company: UCB Pharma S.A
Email: andras.kasa@ucb.com
Website: https://github.com/kasaandras/Yolo_PDF_Table_segmentation

Author Name: Darius Tetsa
Company: UCB Pharma S.A
Email: darius.tetsa@ucb.com