From Breaking Bad to Breaking Data: Unlocking TFL PDFs

Andras Kasa, UCB Pharma, Monheim, Germany
Darius Tetsa Tata, UCB Pharma, Slough, UK

Abstract

Pharmaceutical companies possess extensive data collections that remain inaccessible to machines due to their unstructured format. PDF TFL outputs are a particular example of unstructured data where content and knowledge are disseminated in complex layouts including headers, text body and footers, and are organized over multiple hierarchies. While it has become easy to extract text from PDF documents, it remains difficult to keep track of the structural organisation of the document and the semantics of the content. Our experiments focus on understanding the layout (structural organization) of a standard TFL PDF output and applying content segmentation to extract the knowledge hidden in the document using various techniques, including simple clustering and computer vision. This paper details our experimentations and the outcomes. We propose that enabling our computers to effectively understand the structure and semantics of our TFL outputs could ease the servicing of LLMs in RAGs to improve insight generation and enhance many other use cases such as metadata recommendation, process standardization, automated document generation, and SAS or R program skeleton generation.

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 (Namboodiri & Jain, 2007).

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 reusability and insight generation.

In the following sections, we will explore state-of-the-art techniques we have experimented with and their outcomes.

Background

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. Nonetheless, the UCB TFL PDF format is organized into distinct sections:

- **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.

See below for an example of the Table output PDF layout.

- Fig 1: Example Table output layout

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.

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

Library	Text extraction performance	OCR recognition	TFL Layout recognition
PyPDF2[1]	Good	Good	Poor
PyMuPDF[2]	Good	Good	Poor
pdfminer[3]	Good	Poor	Poor
EasyOCR[4]	Good	Good	Poor

Table 1: Summary of PDF data extraction tools.

From Table 1, we observe that while all the PDF data extraction tools performed well in extracting data from the PDF TFL outputs, none were able to recognize the structure of the outputs. They all showed similar performance in reading the PDF files.

We also noticed inconsistencies in the extraction process. Although the document was usually read line by line, sometimes it was processed from left to right and other times from right to left. Additionally, the libraries were not sophisticated enough to recognize table structures and line breaks.

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 algorithm, as these typically do not require pre-training.

Hierarchical clustering

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 (Nielsen, 2016). 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 is commonly applied in image segmentation, where similar pixels or regions are grouped based on color, texture, or other visual features. It finds applications in fields such as medical imaging, remote sensing, and computer vision.

After applying hierarchical clustering to our TFL outputs, the results were not entirely 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. We concluded that this approach does not necessarily bring us closer to our goal.

[1] https://pypdf2.readthedocs.io/en/3.x/
[2] PyMuPDF 1.25.4 documentation
[3] https://pypi.org/project/pdfminer/
[4] https://pypi.org/project/easyocr/1.1.4/

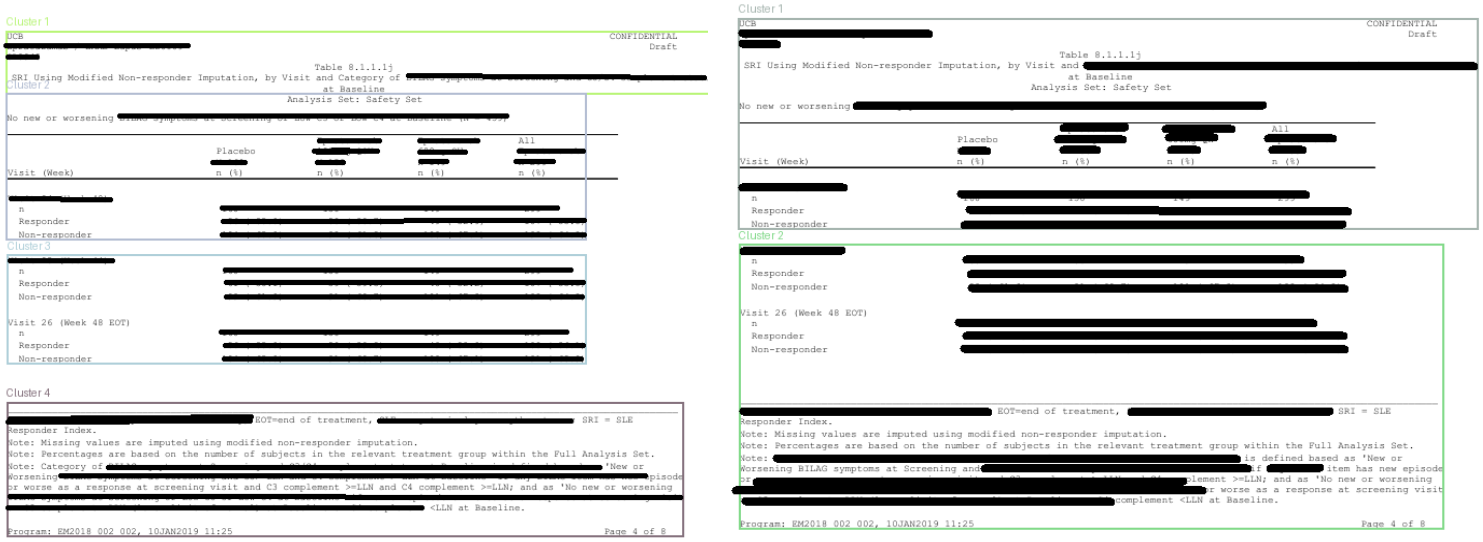

Fig 2: Example of Hierarchical clustering on TFL outputs with 4 vs 2 predicted clusters

Supervised Learning and Computer Vision

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.

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

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 (Terven, Córdova-Esparza, & Romero-González, 2023). 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.

We trained YOLO 8.0 recognize the layout of the TFL outputs.

CVAT Implementation

CVAT (Computer Vision Annotation Tool) is a free, open-source, web-based tool designed for image and video annotation. It supports key supervised machine learning tasks including object detection, image classification, and image segmentation (Christos Moschidis, 2025).

The tool offers advanced annotation capabilities through multiple formats: bounding boxes, polygons, points, skeletons, cuboids, trajectories, and more. This versatility makes CVAT suitable for various computer vision tasks such as classification, tracking, object detection, pose estimation, and attribute tagging.

To maintain focus in our initial experiments, we limited our scope to tables only. For our implementation, we:

1. Annotated approximately 100 Table outputs using four distinct labels: Header, Title, Body, and Footer.
2. Selected a diverse collection of Table 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 Tables outputs into image files compatible with YOLO.
5. Trained YOLO on these annotated images to recognize the different document sections.
6. Validated the trained model by testing its section prediction capabilities on new TFL outputs.

Results:

The model effectively identified the main sections of new PDF outputs with a confidence score of 0.5, demonstrating robust recognition capabilities and creating bounding boxes around each section. It achieved a recall of 96%, with approximately 9 out of 10 detections correct (precision of 0.94), resulting in an overall F1 score of 96.8%.

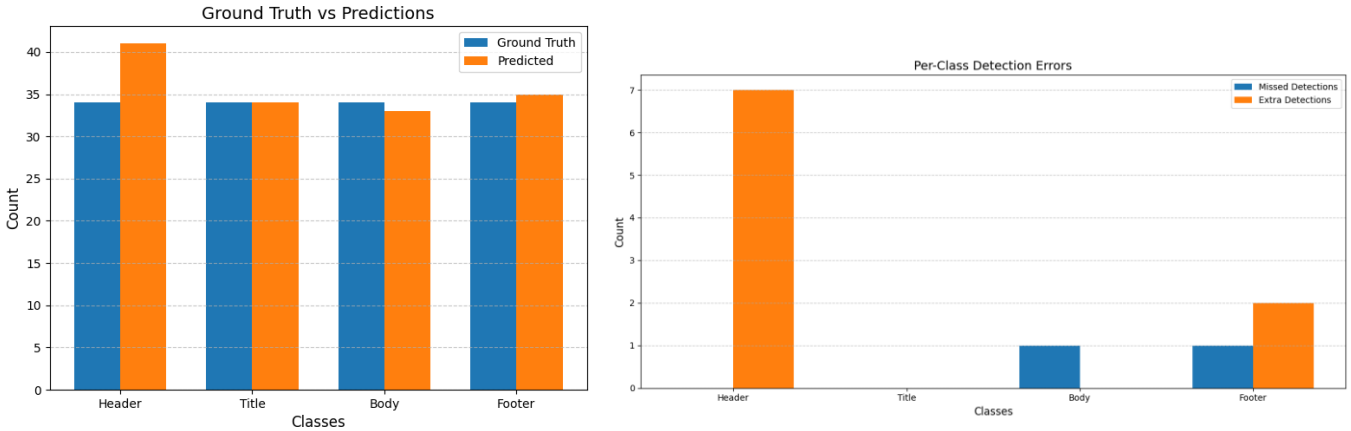

Fig 3: Performance of our model

Below are examples of TFL outputs with section predictions by our model. In Figure 4, the model clearly detects all sections of our output and creates bounding boxes around them. In Figure 5, the model even detects a footer text embedded within the body section.

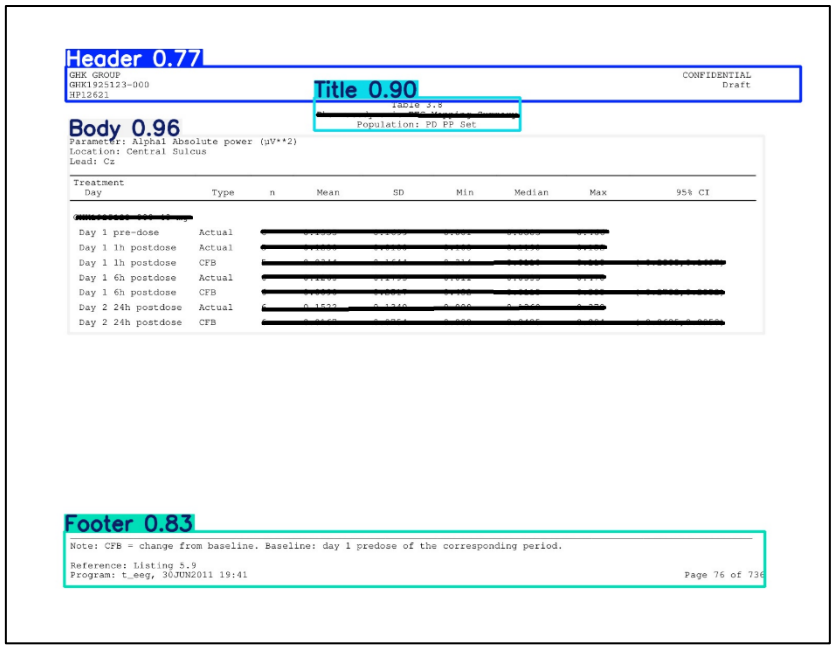

Fig 4: Result of the prediction on a table. All sections clearly labelled by the model.

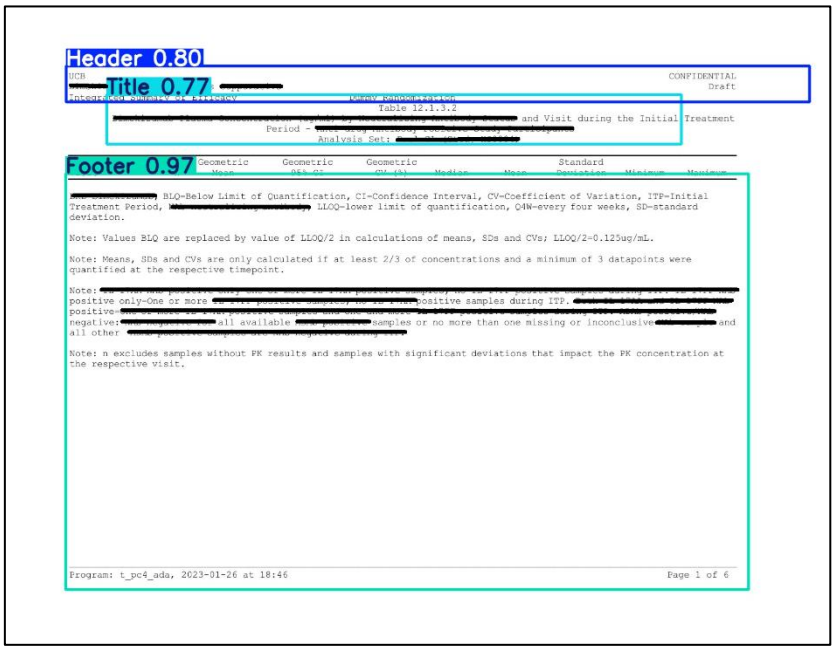

Fig 5: Result of the prediction on a table. Model detects a footer text embedded in the body section.

Discussion

Fine-tuning a trained YOLO model for generating TFL PDF outputs has demonstrated robust performance in accurately identifying the various sections it was trained on. The coordinates of the predicted bounding boxes enable us to extract the content from each section effectively. Moreover, by integrating specialized tools—such as natural language processing techniques and large language models—we can further interpret and analyze the extracted content. For example, NLP methods and LLMs can provide a deeper understanding of the title section, while dedicated tools can be employed to accurately extract and process the body of tables.

For extracting body sections of tables or listings, we recommend using the model developed by Zhang et al. in their work, *Table Separation Line Detection Based on Instance Segmentation*. (Zhang, 2023) This model, created by researchers from the University of Science and Technology of China and iFLYTEK AI Research, offers a specialized approach to table detection that can enhance the accuracy and efficiency of content extraction from complex document layouts.

Moreover, by extracting specific sections of the output, we can seamlessly connect and compare tables across different documents. This capability not only enhances insight generation but also supports robust data-driven decision-making.

A deep understanding of the semantics behind a TFL output layout unlocks numerous use cases, such as semi-automated narrative creation, the construction of compound and company-level knowledge bases, and streamlined handling of authority queries.

Despite these promising results, there remain areas for improvement. One key issue is the limited size of our training data, which restricts the model's ability to further specialize. We can address this by annotating additional TFL outputs and pretraining YOLO with them. We estimate that at least 800 tables and listings, along with around 600 figures, are needed. Additionally, by integrating further deterministic tools, we can enhance the accuracy of the bounding boxes if necessary.

Additionally, our model was trained exclusively on UCB's TFL outputs, which may restrict its generalizability to non-UCB TFL outputs. We would welcome collaborations with other companies willing to share annotated and anonymized outputs, enabling us to refine the model and make it more adaptable and industry-ready. Another challenge is the need to convert PDF files into image files for YOLO to process. This conversion requires additional work to connect the various pages of a multi-page TFL output file. Addressing these challenges will be crucial for enhancing the model's performance, the tools built upon it, and its overall applicability.

Conclusion

Our experiments successfully developed a model that can unlock the structure of Table outputs, facilitating insight generation and data-driven decision-making. By training YOLO to recognize various sections of the outputs, we have opened new opportunities for exploring innovative use cases. This advancement enhances our ability to derive valuable insights from TFL outputs, paving the way for more informed and effective decision-making processes.

Our experiments successfully developed a model that can unlock the structure of table outputs, facilitating insight generation and data-driven decision-making. By training YOLO to recognize various sections of the outputs, we have opened up new opportunities for exploring innovative use cases.

We are also considering a hybrid approach that combines supervised learning with reinforcement learning. This means that once YOLO proposes bounding boxes, we would review the results and correct the boxes when needed, then reinforce YOLO's learning using these reviewed sets.

This advancement enhances our ability to derive valuable insights from table outputs, paving the way for more informed and effective decision-making processes.

Acknowledgement

We would like to thank our management team, and especially Alberto Montironi, Phyllis Semtana, Tim Williams and Mike Branson, for their continuous support.

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.

Bibliography

Christos Moschidis, E. V. (2025). *Annotation tools for computer vision tasks.* Seventeenth International Conference on Machine Vision (ICMV 2024). doi:https://doi.org/10.1117/12.3055065

Namboodiri, A. M., & Jain, A. K. (2007). Document Structure and Layout Analysis. *Springer London*.

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

Terven, J., Córdova-Esparza, D.-M., & Romero-González, J.-A. (2023). *A Comprehensive Review of YOLO Architectures in Computer Vision: From YOLOv1 to YOLOv8 and YOLO-NAS.* (5 ed.). Machine Learning and Knowledge Extraction. doi:https://doi.org/10.3390/make5040083

Zhang, Z. H. (2023). able Separation Line Detection Based on Instance Segmentation. (5th ed.). University of Science and Technology of China;. *iFLYTEK AI Research.* .