

ADAM MICKIEWICZ UNIVERSITY, POZNAŃ
FACULTY OF MATHEMATICS AND COMPUTER SCIENCE



Michał Turski

**Utilizing Structured Resources
in Neural Language Models**

Ph.D. thesis

Thesis supervisor:
Prof. UAM dr hab. Filip Galiński

Discipline:
Computer and Information Sciences

Poznań, 2024

UNIwersytet IM. Adama Mickiewicza w Poznaniu
Wydział Matematyki i Informatyki



Michał Turski

**Wykorzystanie zasobów ustrukturyzowanych
w neuronowych modelach języka**

Rozprawa doktorska

Promotor:
Prof. UAM dr hab. Filip Galiński

Dyscyplina:
Informatyka

Poznań, 2024

Abstract

The majority of research in the field of Natural Language Processing is focused on processing plain text. While this paradigm is highly effective for numerous use cases, such as machine translation, summarization, and chatbots, it fails to fully harness the richness of many texts created by and for humans. Documents, on the other hand, convey meaning not only through their textual content but also through their structure and visual features. A key challenge tackled by this thesis is to develop solutions that combine recent advancements in language modeling with structural information to improve the processing and comprehension of documents.

This thesis comprises five scientific papers in the domain of document understanding, divided into two main sections. The first section focuses on evaluating document understanding models, introducing the first benchmark in this area and proposing a novel dataset in the scientific domain. The proposed benchmark includes a diverse range of document types and tasks, enabling a comprehensive evaluation of document understanding models. The novel dataset, designed specifically for scientific documents, assesses models' ability to reason over both tables and text simultaneously.

The second section of this thesis tackles various challenges in the document understanding domain, proposing innovative solutions to enhance model performance. These include a diverse, multilingual corpus for pretraining document-oriented language models, enabling improved understanding of documents across languages and domains; a novel architecture extending capabilities of Transformer model by using structural information, enhancing its ability to process and comprehend structured documents; and a framework for generating tables using a language model, enabling the creation of structured data from natural language input.

Overall, this thesis contributes to the development of more accurate and useful document understanding models, enabling improved processing and comprehension of rich, structured documents.

Streszczenie

Większość badań w dziedzinie przetwarzania języka naturalnego koncentruje się na przetwarzaniu tekstu. Choć ten paradygmat jest bardzo skuteczny w wielu zastosowaniach, takich jak tłumaczenie maszynowe, automatyczne podsumowywanie i systemy dialogowe, nie potrafi w pełni wykorzystać bogactwa wielu dokumentów tworzonych przez i dla ludzi. Dokumenty przekazują znaczenie nie tylko przez warstwę tekstową, ale także poprzez swoją strukturę i cechy wizualne. Kluczowym wyzwaniem podejmowanym w tej pracy jest proponowanie rozwiązań rozszerzających najnowsze modele języka o wykorzystanie informacji strukturalnych celem poprawy jakości przetwarzania dokumentów.

Niniejsza rozprawa składa się z pięciu prac naukowych w domenie rozumienia dokumentów i jest podzielona na dwie główne sekcje. Pierwsza sekcja dotyczy problemu oceny modeli rozumienia dokumentów, wprowadzając pierwsze wyzwanie (ang. *benchmark*) w tej domenie oraz proponuje nowy zbiór danych oparty na piśmiennictwie naukowym. Zaproponowane wyzwanie obejmuje różnorodny zakres dokumentów i zadań, umożliwiając kompleksową ocenę modeli rozumienia dokumentów. Nowy zbiór danych, zaprojektowany specjalnie dla dokumentów naukowych, ocenia zdolność modeli do rozumienia tekstu z wykorzystaniem tabeli jako dodatkowego źródła informacji.

Druga sekcja tej pracy podejmuje różne wyzwania w domenie rozumienia dokumentów, proponując innowacyjne rozwiązania mające na celu poprawę jakości modeli. Są to: różnorodny, wielojęzyczny korpus do uczenia modeli języka przeznaczonych dla dokumentów, umożliwiający lepsze rozumienie dokumentów w różnych językach i dziedzinach; nowa architektura rozszerzająca model Transformer o kodowanie informacji strukturalnych, co pozwala na przetwarzanie dokumentów o bogatej strukturze; oraz metoda generowania tabel przy użyciu modelu języka, umożliwiająca tworzenie strukturalnych danych z wejścia w postaci tekstu.

Podsumowując, ta praca przyczynia się do rozwoju modeli rozumienia dokumentów, umożliwiając lepsze przetwarzanie i analizę dokumentów o bogatej strukturze.

Contents

Contents	ix
1 Foreword	1
1.1 Motivation	1
1.2 Structure and Scope of Thesis	3
1.2.1 Measuring state of document understanding	4
1.2.2 2D-Structured Language Modeling	5
1.3 List of publications	7
1.4 The Impact of My PhD Work	8
1.4.1 Contribution to the development of the field	8
1.4.2 Industrial impact	9
References	9
MEASURING STATE OF DOCUMENT UNDERSTANDING	11
2 DUE: End-to-End Document Understanding Benchmark	13
3 Arxiv Tables: Document Understanding Challenge Linking Texts and Tables	39
2D-STRUCTURED LANGUAGE MODELING	54
4 CCpdf: Building a High Quality Corpus for Visually Rich Documents from Web Crawl Data	57
5 LAMBERT: Layout-Aware Language Modeling for Information Extraction	76
6 STable: Table Generation Framework for Encoder-Decoder Models	93
APPENDIX	113
A Declarations of Contribution	115
A.1 DUE: End-to-End Document Understanding Benchmark Contributions Declaration	116
A.2 Arxiv Tables: Document Understanding Challenge Linking Texts and Tables Contributions Declaration	118
A.3 LAMBERT: Layout-Aware Language Modeling for Information Extraction Contributions Declaration	119
A.4 CCpdf: Building a High Quality Corpus for Visually Rich Documents from Web Crawl Data Contributions Declaration	120
A.5 STable: Table Generation Framework for Encoder-Decoder Models Contributions Declaration	121
B Awards, Patents, and Projects	123
B.1 IAPR/ICDAR 2021 Best Industry Related Paper Award	123
B.2 Encoder-decoder transformer for table generation patent	123
B.3 Projects	124

1.1 Motivation

Natural Language Processing (NLP) tasks, such as sentiment analysis, machine translation, information retrieval, and text summarization, heavily rely on text-based representations and encoding techniques to effectively capture the semantic, syntactic, and contextual information within textual data. Most traditional tasks (such as the ones listed before) operate on plain text. Nonetheless, the increasing understanding of the inherent multimodal nature of human communication and the associated challenges has given rise to the emergence of multimodal tasks. These tasks leverage not only textual data but also additional modalities such as images [1], audio [2], videos [3], or entire documents, including their structure: layout and visual features [4].

Among the modalities listed, entire documents are particularly noteworthy for two reasons. First, such documents are abundant and available across a wide range of domains, including news articles, research papers, legal documents, and technical manuals. The majority of processing for these documents is currently done manually, making automation a promising avenue for improving productivity. By tackling document understanding challenges, multimodal language models can unlock new possibilities in areas like information retrieval, virtual assistants, and content understanding. The intelligent document processing market is experiencing rapid growth, with its size estimated at USD 1.1 billion in 2022 and projected to reach USD 5.2 billion by 2027 [5].

Secondly, the problems associated with multimodal documents present novel research challenges. Understanding the semantics and relationships within the document structure requires multimodal models to effectively integrate textual, spatial, and visual information. In the NLP domain, most state-of-the-art models employ the Transformer architecture, which initially utilized trigonometry-based positional embeddings to represent the order of tokens [6]. More recent research has explored alternative methods for encoding token order, such as the introduction of relative positional biases in T5 [7] and the use of Rotary Embeddings [8]. All of these solutions share the common approach of representing the structure of text as an ordered list of tokens. While this representation is suitable for traditional NLP tasks, it results in significant information loss in the domain of structure-rich documents.

Figure 1.1 displays a page from a document with different structures highlighted. The conventional approach would involve preprocessing this page using an Optical Character Recognition (OCR) tool. However, during this process, structural information is lost. Figure 1.2 presents a comparison between the top part of the page and its output after being processed by an OCR tool. The figure also displays the reading order generated by the OCR tool. Using plain text produced by OCR tool as a model input may be confusing in some cases. For example, consider the fragment of the output generated by an OCR tool: "Website: The

1.1	Motivation	1
1.2	Structure and Scope of Thesis	3
1.2.1	Measuring state of document understanding . . .	4
1.2.2	2D-Structured Language Modeling	5
1.3	List of publications	7
1.4	The Impact of My PhD Work	8
1.4.1	Contribution to the development of the field	8
1.4.2	Industrial impact	9
	References	9

plan charge as shown in Tables 1-3 is payable when the application is deposited. www.buildingcontrolpartnershiphants.gov.uk/". This output significantly deviates from the original meaning conveyed by the two-column layout of the document.

In response to these challenges, there has been a surge of research in recent years on document-focused language models. Notable examples include LayoutLM [9, 10], TILT [11], DocFormer [12], and UDOP [13]. These models share a common goal: leveraging document structure to enhance the processing of semantic information. To achieve this, all of them incorporate some form of structural positional encoding, with some models also utilizing the visual layer of a document.

BUILDING CONTROL CHARGES
Charges with effect from 1 January 2010 for applications within **Gosport Borough or Fareham Borough Councils**.
Please make cheques payable to Fareham Borough Council.

There are three tables of charges:
Table 1 NEW DWELLINGS
Table 2 SMALL BUILDINGS, EXTENSIONS or ALTERATIONS
Table 3 ANY OTHER WORK

After determining which table the work your proposal falls into, you may calculate the charge payable. Where multiple works are proposed, i.e. extension and internal alterations, the separate charges should be added together. There are two methods of payment depending on the type and scale of work, i.e.

FULL PLAN APPLICATIONS
The **plan** charge as shown in Tables 1-3 is payable when the application is deposited. We will send you an invoice for the **inspection** charge after works commence.

BUILDING NOTICE APPLICATIONS
Under all tables the whole charge (plan plus inspection charge added together) is payable when an application is deposited.

GENERAL
Value added tax is payable on all applications except for regularisations. (VAT inclusive figures are shown in brackets in Tables 1, 2 & 3). Where a regularisation application is submitted the charge will be 120% of that stated in Tables 1, 2 & 3.
The application may be exempt from charges if it is for work necessary to improve facilities for a disabled person, subject to certain criteria being met. A disabled person, whether registered or registrable, should supply supporting information with their Building Regulation application.

TABLE 1
NEW DWELLINGS Up to 300m² and not exceeding 3 stories (VAT Inclusive figure in brackets)

Number of dwellings	PLAN CHARGE		INSPECTION CHARGE	
	Charge £	Additional dwellings	Charge £	Additional dwellings
1	150.00 (176.25)		377.00 (442.98)	
2	205.00 (240.88)		522.00 (613.35)	
3	278.10 (326.77)		682.89 (802.40)	
4	348.40 (409.37)		797.00 (936.48)	
5	430.00 (505.25)		899.00 (1056.33)	
6	504.00 (592.20)		1011.00 (1187.93)	
7	525.00 (616.88)		1080.00 (1269.00)	
8	546.00 (641.55)		1259.00 (1479.33)	
9	568.00 (667.40)		1437.36 (1688.90)	
10	573.00 (673.28)		1633.46 (1919.32)	
11	578.00 (679.15)		1790.00 (2103.25)	
12	583.00 (685.03)		1945.00 (2285.38)	
13	589.00 (692.08)		2101.00 (2468.68)	
14	594.00 (697.95)		2230.00 (2620.25)	
15	599.00 (703.83)		2384.00 (2801.20)	
16	605.00 (710.88)		2539.00 (2983.33)	
17	610.00 (716.75)		2694.00 (3165.45)	
18	615.00 (722.63)		2849.00 (3347.58)	
19	621.00 (729.68)		2968.00 (3487.40)	
20	626.00 (735.55)		3121.00 (3667.18)	
21 to 30	636.00 (747.30)	£10.60 (£12.46) per dwelling exceeding 21	3192.00 (3750.60)	£98.47 (£115.70) per dwelling exceeding 21
31 and over	742.00 (871.85)	£5 (£5.88) per dwelling exceeding 31	4176.70 (4907.62)	£75 (£88.13) per dwelling exceeding 31

If you are not sure about any aspect of your project, the Building Control Partnership will be pleased to discuss any query and provide a quote for your Building Regulation Application.

Figure 1.1: Example page from the CCpdf corpus featuring a structure-rich document. Color boxes have been used to highlight the different structures present in the document.

Source of the document: buildingcontrolpartnershiphants.gov.uk.

BUILDING CONTROL CHARGES

Charges with effect from 1 January 2010 for applications within **Gosport Borough or Fareham Borough Councils**.

Please make cheques payable to Fareham Borough Council.

There are three tables of charges:

- Table 1 NEW DWELLINGS
- Table 2 SMALL BUILDINGS, EXTENSIONS or ALTERATIONS
- Table 3 ANY OTHER WORK

After determining which table the work your proposal falls into, you may calculate the charge payable. Where multiple works are proposed, i.e. extension and internal alterations, the separate charges should be added together. There are two methods of payment depending on the type and scale of work, i.e.

FULL PLAN APPLICATIONS

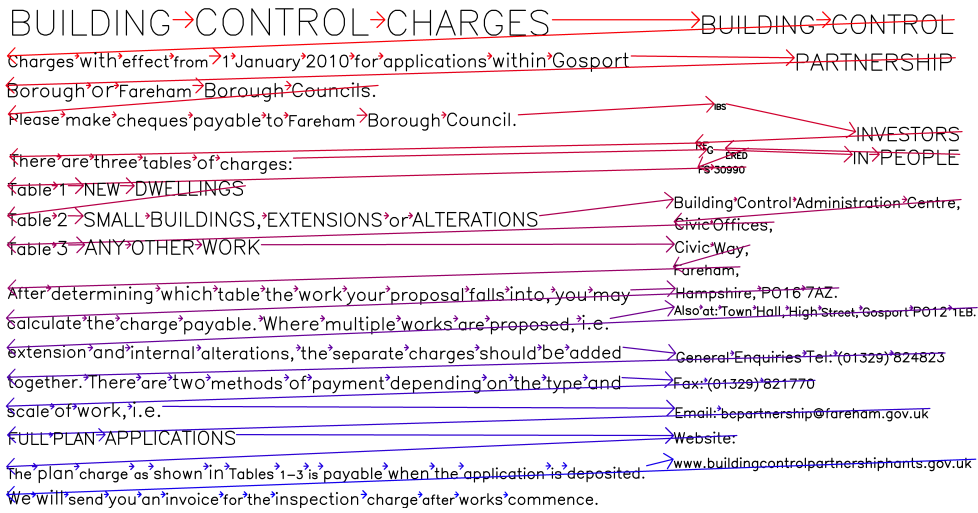
The **plan** charge as shown in Tables 1-3 is payable when the application is deposited. We will send you an invoice for the **inspection** charge *after* works commence.



**Building Control Administration Centre,
Civic Offices,
Civic Way,
Fareham,
Hampshire, PO16 7AZ.**
Also at: Town Hall, High Street, Gosport PO12 1EB.

General Enquiries Tel: (01329) 824823
Fax: (01329) 821770
Email: bcpartnership@fareham.gov.uk
Website: www.buildingcontrolpartnershiphants.gov.uk

(a) Page fragment



(b) OCR output with reading order marked

Figure 1.2: Comparison between document and OCR tool output

The primary objectives of this thesis are twofold: first, to measure the state-of-the-art in the domain of structure-rich document understanding by evaluating the performance of existing models; and second, to improve existing solutions through the development and evaluation of novel approaches that enhance the accuracy, efficiency, and usability of document understanding tools.

1.2 Structure and Scope of Thesis

The research presented in this thesis adheres to the standard experimental methodology employed in the machine learning field. The process involves four main steps: defining a set of problems to be addressed, identifying datasets that address these problems, proposing solutions, and conducting experimental validation of proposed solutions. Similarly,

the structure of this thesis follows this methodology: I begin by defining and motivating the problem of language modeling for structure-rich documents, introduce evaluation datasets, discuss the preparation of pretraining data, and finally present and evaluate model architectures designed to tackle various document understanding problems.

This thesis is composed of five papers that all focus on the domain of understanding documents that have complex structure. The first two papers are dedicated to measuring the current state of document understanding, while the remaining three papers propose various improvements to existing solutions. These improvements fall into three main categories: (1) data preparation, (2) input structure representation, and (3) generating structured output from a model.

1.2.1 Measuring state of document understanding

DUE: End-to-End Document Understanding Benchmark

Chapter 2 introduces a benchmark for measuring the current state of document understanding. The work was presented at the NeurIPS 2021 conference. This benchmark is composed of seven datasets that were carefully selected based on several criteria, including: (1) quality, (2) difficulty, (3) licensing, and (4) similarity to real-world use cases of document understanding systems. In order to ensure that these criteria are met as closely as possible, some of the datasets were cleaned, re-annotated, and reformulated. One of my contributions was further enriching the datasets with diagnostic annotations, which provide valuable insights into the strengths and weaknesses of each submitted model.

The documents included in the benchmark, such as tables, forms, or infographics, have a complex structure that necessitates a model with the capacity to understand them. Furthermore, for certain datasets, the expected output format is also intricate, necessitating the model to produce a list or a table.

Another significant outcome of this work is the introduction of a new dataset representation format, which enables the presentation of various document understanding tasks in a consistent and unified manner. As a result, all datasets included in the challenge have been converted to this standardized format. Moreover, competitive baseline models were proposed, and all of the datasets and baselines were made publicly available.

Arxiv Tables: Document Understanding Challenge Linking Texts and Tables

Chapter 3 presents dataset for understanding a text sample in the context of the accompanying table. The data comes from the scientific domain and contains domain nomenclature and special characters (such as Greek letters and scientific symbols), which are challenging for state-of-the-art models. The proposed dataset is the largest publicly available dataset in the area of table understanding. In addition, I proposed baseline models for others to compare their solutions, and both the dataset and these baselines have been made accessible to the public. The work was

presented at VINALDO: Machine vision and NLP for Document Analysis workshop at the ICDAR 2023 conference.

1.2.2 2D-Structured Language Modeling

CCpdf: Building a High Quality Corpus for Visually Rich Documents from Web Crawl Data

Chapter 4 includes work presented at the ICDAR 2023 conference. The focus of this work is on the preparation of a large-scale, diverse, multilingual, publicly available corpus for pretraining document understanding models. Previous corpora used for this purpose were either limited to a single domain and monolingual or were not publicly available. This work aims to address this gap by providing the research community with a corpus that was created using a carefully designed pipeline.

The source of documents in this dataset is Common Crawl, an open repository of web data. The proposed pipeline consists of four key components: (1) document detector responsible for identifying and extracting documents from the raw web data, (2) language detector utilized to determine the language used in each extracted document, (3) diversity-oriented filters implemented to ensure a diverse range of documents are included in the dataset, avoiding excessive redundancy, (4) OCR tools employed to convert scanned or image-based documents into machine-readable text.

The pipeline design decisions were informed by a comparative evaluation of different alternative methods, and the impact of each method was measured using metrics such as data quality, quantity, processing time, scalability, and costs. Furthermore, the shared corpus of documents underwent a comprehensive analysis and description. On one hand, this analysis provides valuable insights into the training data, enabling a deeper understanding of the models trained on it. On the other hand, it uncovers intriguing properties of the documents that are accessible on the Internet, shedding light on the characteristics and diversity of the web content. The resulting corpus, along with accompanying metadata, has been made publicly available.

LAMBERT: Layout-aware language modeling for information extraction

Chapter 5 introduces a novel method for modeling documents with intricate structure, where non-trivial layout aspects have a significant impact on semantics. This innovative approach modifies the encoding of input token positions to reflect their corresponding locations on a page. This modification is applied to a pre-trained Transformer model without altering its existing weights, resulting in a more efficient adaptation training process compared to full pre-training. Notably, adaptation training is considerably shorter and requires less data.

Two distinct methods are employed to incorporate layout positional information into the model. Firstly, the embeddings are enhanced by adding terms based on the position of each token on the page. Secondly, a 2D relative bias is introduced into the attention weights, which is

contingent on the vertical and horizontal distance between two tokens. As a result of these modifications, the model surpasses the state-of-the-art on four datasets.

A thorough ablation study compares four distinct methods of representing the structure of a document and their combinations. Two of these methods solely encode the sequential positions of tokens, while the other two incorporate layout positional information, as described in the previous paragraph.

The work proposed one of the attempts to apply the NLP model in the document understanding domain. The manuscript was submitted for review just a few days prior to the publication of the preprint for LayoutLM [9]. The paper was presented at the ICDAR 2021 conference and received the IAPR/ICDAR 2021 Best Industry Related Paper Award.

STable: Table Generation Framework for Encoder-Decoder Models

Chapter 6 proposes a novel method for generating output in the structure of a table from a language model. A table is a unique form of knowledge representation easy to read by humans as well as easy to process by computers. Most language models produce their output in the structure of plain text. The simplest way to generate a table by such a model is to generate it in natural reading order: left to right, top to bottom. Unfortunately, such an approach is prone to errors: after a first mistake model tends to get lost in the structure of a table and quality falls rapidly. To avoid this issue STable model generates a table in a confidence-driven way: instead of filling a table left to right and top to bottom, it estimates the probability of making an error in each cell and fills a cell with the lowest probability. Then it iteratively repeats the process until the whole table is filled.

The model must undergo special training to generate a table in this way. Therefore, the work proposes a training procedure in which, in each epoch, the model is trained to predict the same table in a different order of cells. Thanks to that, the model is capable of generating any cell at each iteration. In addition, I propose a pretraining task and dataset that enable the model to learn faster on downstream tasks and require fewer training examples.

Positional encoding in the decoder was also modified. As each generated token's position within the tabular structure is known, the model uses this information to encode the token's position. Encoding operates on multiple levels: it depends on the token's row, column, and position within a particular cell.

The text-to-table paradigm employed in this paper can be extended beyond datasets with anticipated tabular outputs to other NLP tasks, including joint entity and relation recognition and complex information extraction. Consequently, the proposed model has the potential to be applied to a diverse range of tasks that have traditionally been addressed using specialized models." The proposed model was better than left to right top to bottom approach on 5 out of 8 diverse datasets. It beat state-of-the-art on 1 public and 3 private datasets. The paper was presented at EACL 2024 conference.

1.3 List of publications

Table 1.1: List of publications included in the thesis.

Title	Authors	Volume	Points [†]
DUE: End-to-End Document Understanding Benchmark	Ł. Borchmann*, M. Pietruszka*, T. Stanisławek*, D. Jurkiewicz, M. Turski, K. Szyndler, and F. Galiński	<i>Advances in Neural Information Processing Systems 34</i> (NeurIPS 2021)	200
Arxiv Tables: Document Understanding Challenge Linking Texts and Tables	K. Konopka, M. Turski, and F. Galiński	<i>Document Analysis and Recognition – ICDAR 2023 Workshops</i> (ICDAR 2023)	0 or 140 [‡]
CCpdf: Building a High Quality Corpus for Visually Rich Documents from Web Crawl Data	M. Turski, T. Stanisławek, K. Kaczmarek, P. Dyda, and F. Galiński	<i>Document Analysis and Recognition – ICDAR 2023</i> (ICDAR 2023)	140
LAMBERT: Layout-aware language modeling for information extraction	Ł. Garncarek*, R. Powalski*, T. Stanisławek*, B. Topolski*, P. Halama, M. Turski, and F. Galiński	<i>Document Analysis and Recognition – ICDAR 2021</i> (ICDAR 2021)	140
STable: Table Generation Framework for Encoder-Decoder Models	M. Pietruszka*, M. Turski*, Ł. Borchmann*, T. Dwojak, G. Nowakowska, K. Szyndler, D. Jurkiewicz, and Ł. Garncarek	<i>Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics</i> (EACL 2024)	140

The thesis is based on five publications that were published between 2021 and 2024. Four of these publications were presented at international conferences, while the fifth was presented at a workshop that accompanied such a conference. A complete list of the publications can be found in Table 1.1 and Table 1.2 presents my individual contributions to each of the publications. Appendix A present declarations that outline the individual contributions made by each author.

* equal contribution

† Ministerstwo Edukacji i Nauki (Ministry of Science and Higher Education) points

‡ Ambiguous valuation; Workshops proceedings were published as a separate volume.

Table 1.2: My contributions to the publications included in the thesis.

Title	My contibution
DUE: End-to-End Document Understanding Benchmark	Taxonomy of diagnostic labels, methodology, organization, and control of human annotation process, organization, and control of measuring human performance, editing the paper
Arxiv Tables: Document Understanding Challenge Linking Texts and Tables	Conceptualization and implementation of proposed baseline solutions, running experiments, writing and editing the paper
CCpdf: Building a High Quality Corpus for Visually Rich Documents from Web Crawl Data	Conceptualization and methodology, design and implementation of the tool for managing the corpus, statistical analyses and exploration of the corpus, running proposed pipeline and sharing the data, writing and editing the paper, project leadership
LAMBERT: Layout-aware language modeling for information extraction	Creating final dataset for model training with methodology for filtering out non business/legal documents, implementation of training flow for training LAMBERT models, editing the paper
STable: Table Generation Framework for Encoder-Decoder Models	Conceptualization and methodology of the research work, idea behind pretraining, preparation of domain-specific pretraining dataset, data preprocessing and postprocessing, baselines implementation, running pretraining, experiments, and ablation studies, error analysis, writing the paper, project leadership

1.4 The Impact of My PhD Work

During my Ph.D. (2020-2024), the field of document understanding underwent substantial growth, both in terms of research progress and adaptation by the industry. On one hand, the advent of generalist multimodal large language models dramatically transformed the landscape. On the other hand, there was a significant increase in the number of models and datasets specifically tailored for document understanding tasks. In this chapter, I will provide a concise summary of how my research contributions played a role in advancing the field during this period of rapid progress.

1.4.1 Contribution to the development of the field

I would like to emphasize the significant impact of two of my publications.

Firstly, my paper titled *LAMBERT: Layout-aware language modeling for information extraction* was one of the pioneering structure-oriented language models in the field (along with LayoutLM [9]). It not only inspired TILT [11], the first generative model for document understanding, but also influenced a multitude of subsequent works, including UDOP [13]. Furthermore, this publication has been highly cited, with a total of 131 citations to date. Additionally, it received the prestigious IAPR/ICDAR 2021 Best Industry Related Paper Award in recognition of its substantial impact on the field of document understanding.

Secondly, my publication titled *DUE: End-to-End Document Understanding Benchmark* has had a significant impact on the field of document understanding as well. Numerous authors have utilized the publicly released data from this benchmark to train and evaluate their solutions, resulting in a total of 35 citations. Furthermore, the collaborative network that was established during the development of this benchmark has facilitated cross-institutional collaboration and has led to several important outcomes. These include the organization of the DUDE (Document Understanding of Everything) competition [14] at the ICDAR 2023 conference, as well as the publishing of the large-scale DUDE dataset [15]. These developments have further propelled the field of document understanding and have provided a valuable resource for researchers and practitioners alike.

1.4.2 Industrial impact

The work conducted during my Ph.D. was driven by real-world industrial problems and its outcomes have found practical application in production systems. The CCpdf corpus served as a training dataset for models utilized by companies enhance their business processes. Additionally, the LAMBERT model, developed as part of my research, has been successfully deployed in business settings to extract crucial information from documents. Moreover, STable, a key component of my research, has been patented (more details in Appendix B.2) and is currently in the process of being commercialized. I also actively participated in two projects that were implemented as part of the Smart Growth Project, a program funded by the European Union with the aim of promoting research, development, and collaboration between academia and industry in Poland. Appendix B provides a comprehensive list of all the projects and patents in which I have been involved during the course of my Ph.D. studies.

References

- [1] Stanislaw Antol et al. ‘VQA: Visual Question Answering’. In: *International Conference on Computer Vision (ICCV)*. 2015 (cited on page 1).
- [2] Konstantinos Drossos, Samuel Lipping, and Tuomas Virtanen. ‘Clotho: an Audio Captioning Dataset’. In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. 2020, pp. 736–740. DOI: [10.1109/ICASSP40776.2020.9052990](https://doi.org/10.1109/ICASSP40776.2020.9052990) (cited on page 1).
- [3] Zhou Yu et al. ‘ActivityNet-QA: A Dataset for Understanding Complex Web Videos via Question Answering’. In: *AAAI*. 2019, pp. 9127–9134 (cited on page 1).
- [4] Minesh Mathew et al. ‘DocVQA: A Dataset for VQA on Document Images’. In: *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)* (2020), pp. 2199–2208 (cited on page 1).

- [5] MarketsandMarkets Research Private Ltd. *Intelligent document processing market size, share and global market forecast to 2027*. https://web.archive.org/web/20240410181354/https://www.marketsandmarkets.com/Market-Reports/intelligent-document-processing-market-195513136.html?gad_source=1. Accessed: 2024-04-10 (cited on page 1).
- [6] Ashish Vaswani et al. 'Attention is All you Need'. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017 (cited on page 1).
- [7] Colin Raffel et al. 'Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer'. In: *Journal of Machine Learning Research* 21.140 (2020), pp. 1–67 (cited on page 1).
- [8] Jianlin Su et al. 'RoFormer: Enhanced Transformer with Rotary Position Embedding'. In: *ArXiv abs/2104.09864* (2021) (cited on page 1).
- [9] Yiheng Xu et al. 'LayoutLM: Pre-training of Text and Layout for Document Image Understanding'. In: *KDD*. 2020 (cited on pages 2, 6, 8).
- [10] Yang Xu et al. 'LayoutLMv2: Multi-modal Pre-training for Visually-Rich Document Understanding'. In: *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics (ACL) 2021*. 2021 (cited on page 2).
- [11] Rafał Powalski et al. 'Going full-tilt boogie on document understanding with text-image-layout transformer'. In: *Document Analysis and Recognition—ICDAR 2021: 16th International Conference, Lausanne, Switzerland, September 5–10, 2021, Proceedings, Part II* 16. Springer. 2021, pp. 732–747 (cited on pages 2, 8).
- [12] Srikanth Appalaraju et al. 'DocFormer: End-to-End Transformer for Document Understanding'. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. Oct. 2021, pp. 993–1003 (cited on page 2).
- [13] Zineng Tang et al. 'Unifying Vision, Text, and Layout for Universal Document Processing'. In: *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)* (2022), pp. 19254–19264 (cited on pages 2, 8).
- [14] Jordy Van Landeghem et al. 'ICDAR 2023 Competition on Document Understanding of Everything (DUDE)'. In: *Proceedings of the ICDAR 2023*. 2023 (cited on page 9).
- [15] Jordy Van Landeghem et al. 'Document Understanding Dataset and Evaluation (DUDE)'. In: *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*. 2023, pp. 19471–19483. doi: [10.1109/ICCV51070.2023.01789](https://doi.org/10.1109/ICCV51070.2023.01789) (cited on page 9).

**MEASURING STATE OF DOCUMENT
UNDERSTANDING**

DUE: End-to-End Document Understanding Benchmark

Łukasz Borchmann^{*N†}

Michał Pietruszka^{*N‡}

Tomasz Stanisławek^{*N§}

Dawid Jurkiewicz^{N¶}

Michał Turski^{N¶}

Karolina Szyndler^N

Filip Graliński^{N¶}

^NApplica.ai

firstname.lastname@applica.ai

[†]Poznan University of Technology

[‡]Jagiellonian University, Krakow

[§]Warsaw University of Technology

[¶]Adam Mickiewicz University, Poznan

Abstract

Understanding documents with rich layouts plays a vital role in digitization and hyper-automation but remains a challenging topic in the NLP research community. Additionally, the lack of a commonly accepted benchmark made it difficult to quantify progress in the domain. To empower research in this field, we introduce the Document Understanding Evaluation (DUE) benchmark consisting of both available and reformulated datasets to measure the end-to-end capabilities of systems in real-world scenarios. The benchmark includes Visual Question Answering, Key Information Extraction, and Machine Reading Comprehension tasks over various document domains and layouts featuring tables, graphs, lists, and infographics. In addition, the current study reports systematic baselines and analyzes challenges in currently available datasets using recent advances in layout-aware language modeling. We open both the benchmarks and reference implementations and make them available at <https://duebenchmark.com> and <https://github.com/due-benchmark>.

1 Introduction

While mainstream Natural Language Processing focuses on plain text documents, the content one encounters when reading, e.g., scientific articles, company announcements, or even personal notes, is seldom plain and purely sequential. In particular, the document’s visual and layout aspects that guide our reading process and carry non-textual information appear to be an essential aspect that requires comprehension. These layout aspects, as we understand them, are prevalent in tasks that can be much better solved when given not only text sequence on the input but pieces of multimodal information covering aspects such as text-positioning (i.e. location of words on the 2D plane), text-formatting (e.g., different font sizes, colors), and graphical elements (e.g., lines, bars, presence of figures) among others. Over the decades, systems dealing with document understanding developed an inherent aspect of multi-modality that nowadays revolves around the problems of integrating visual information with spatial relationships and text [36, 2, 50, 13].

*Equal contribution

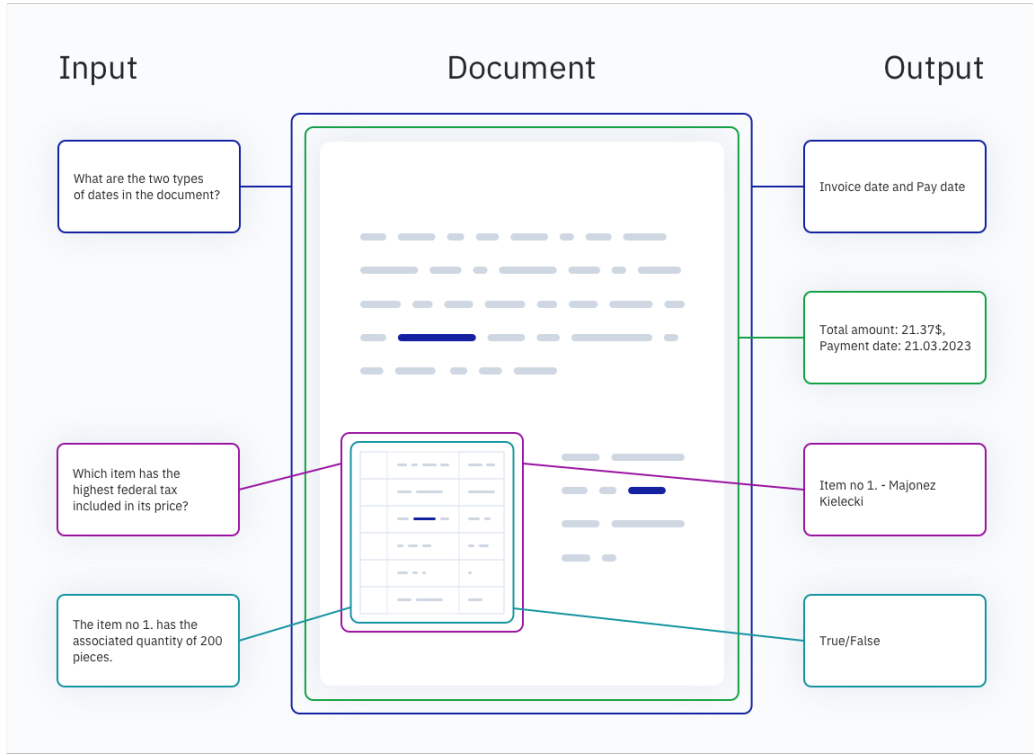


Figure 1: Document Understanding covers problems ranging from the extraction of key information, through verification statements related to rich content, to answering open questions regarding an entire file. It may involve the comprehension of multi-modal information conveyed by a document.

In general, when document processing systems are considered, the term *understanding* is thought of specifically as the capacity to convert a document into meaningful information [10, 57, 16]. This fits into the rapidly growing market of hyperautomation-enabling technologies, estimated to reach nearly \$600 billion in 2022, up 24% from 2020 [42]. Considering that unstructured data is orders of magnitude more abundant than structured data, the lack of tools necessary to analyze unstructured data and extract structured information can limit the performance of these intelligent services. The process of structuring data and content must be robust to various document domains and tasks.

Despite its importance for digital transformation, the problem of measuring how well available models obtain information from a wide range of tasks and document types and how suitable they are for freeing workers from paperwork through process automation is not yet addressed. Meanwhile, in other research communities, there are well-established progress measuring methods, like the most recognizable NLP benchmarks of GLUE and SuperGLUE covering a wide range of problems related to plain-text language understanding [53, 52] or VTAB and ImageNet in the computer vision domain [59, 11]. We intend to bridge this major gap by introducing the first Document Understanding benchmark (available at <https://duebenchmark.com>).

It includes tasks that either originally had a vital layout understanding component or were reformulated in such a way that after our modification, they require layout understanding. In particular, there is no structured representation of the underlying text, such as a database-like table given in advance, and it has to be determined from the input file as a part of the end-to-end process. Every time, there is only a PDF file provided as an input. Additionally, for the convenience of other researchers, we provide information about textual tokens and their locations (bounding boxes) which are coming from the OCR system or directly from the born-digital PDF file (see Section 4).

Contribution. The idea of the paper is to gather, reformulate and unify a set of intuitively dissimilar tasks that we found to share the same underlying requirement of understanding layout concepts. In order to organize them in a useful benchmark, we contributed by performing the following steps:

1. We reviewed and selected the available datasets. Additionally, we reformulated three tasks to a document understanding setting and obtained original documents for all of them (PWC, WTQ, TabFact).
2. We performed data cleaning, including the improvements of data splits (DeepForm, WTQ), data deduplication, manual annotation (PWC, DeepForm), and converted data to a unified format (all datasets).
3. We implemented competitive baselines and measured human performance where it was required (PWC, DeepForm, WTQ).
4. We identified challenges related to the current progress in the DU domain’s tasks and provided manually annotated diagnostic sets (all datasets).

These contributions are organized and described in Table 2. Additionally, a wider review of available tasks is described in Appendix A.

2 The state of Document Understanding

We treat Document Understanding as an umbrella term covering problems of Key Information Extraction, Classification, Document Layout Analysis, Question Answering, and Machine Reading Comprehension whenever they involve rich documents in contrast to plain texts or image-text pairs (Figure 1).

In addition to the problems strictly classified as Document Understanding, several related tasks can be reformulated as such. These provide either text-figure pairs instead of real-world documents or parsed tables given in their structured form. Since both can be rendered as synthetic documents with some loss of information involved, they are worth considering bearing in mind the low availability of proper Document Understanding tasks.

2.1 Landscape of Document Understanding tasks

KIE. Key Information Extraction, also referred to as Property Extraction, is a task where tuple values of the form (property, document) are to be provided. Contrary to QA problems, there is no question in natural language but rather a phrase or keyword, such as *total amount*, or *place of birth*. Public datasets in the field include extraction performed on receipts [20, 38], invoices, reports [45], and forms [24]. Documents within each of the mentioned tasks are homogeneous, whereas the set of properties to extract is limited and known in advance – in particular, the same type-specific property names appear in both test and train sets. In contrast to Name Entity Recognition, KIE typically does not assume that token-level annotations are available, and may require normalization of values found within the document.

Classification. Classification in our context involves rich content, where comprehension of both visual and textual aspects is required since unimodal models underperform. Though document image classification was initially approached using solely the methods of Computer Vision, it has recently become evident that multi-modal models can achieve significantly higher accuracy [55, 56, 40]. Similar conclusions were recently reached in other tasks, e.g., assigning labels to excerpts from biomedical papers [54].

DLA. Document Layout Analysis, performed to determine a document’s components, was initially motivated by the need to optimize storage and the transmission of large information volumes [36]. Even though its motivation has changed over the years, it is rarely an end in itself but rather a means to achieve a different goal, such as improving OCR systems. A typical dataset in the field assumes detection and classification of page regions or tokens [61, 30].

QA and MRC. At first glance, Question Answering and Machine Reading Comprehension over Documents is simply the KIE scenario where a question in natural language replaced a property name. More differences become evident when one notices that QA and MRC involve an open set of questions and various document types. Consequently, there is pressure to interpret the question and

to possess better generalization abilities. Furthermore, a specific content to analyze demands a much stronger comprehension of visual aspects, as the questions commonly relate to figures and graphics accompanying the formatted text [33, 32, 49].

QA over figures. Question Answering over Figures is, to some extent, comparable with QA and MRC over documents described above. The difference is that a ‘document’ here consists of a single born-digital plot, reflecting information from chosen, desirably real-world data. Since questions in this category are typically templated and figures are synthetically generated by authors of the task, datasets in this category contain as many as millions of examples [34, 4].

QA and NLI over tables. Question Answering and Natural Language Inference over Tables are similar, though in the case of NLI, there is a statement to verify instead of a question to answer. There is never a need to analyze the actual layout, as both assume comprehension of a provided data structure in a way that is equivalent to a database table. Consequently, the methods proposed here are distinct from those used in Document Understanding [39, 7].

2.2 Gaps and mistakes in Document Understanding evaluation

Currently available datasets and previous work in the field cannot on their own provide enough information that would allow researchers to generalize results to other tasks within the Document Understanding paradigm. It is crucial to validate models on many tasks with a variety of characteristics a Document Understanding system may encounter in real-world applications. Notably, the scope of the challenges in a single dataset is limited to a specific task (e.g., Key Information Extraction, Question Answering) or to a particular (sub)problem (e.g., processing long documents in Kleister [45], layout understanding in DocBank [30]).

Simultaneously, a common practice in the community is to evaluate models on private data [27, 12, 37, 31] or task-specific datasets selected by authors independently [55, 56, 63, 40, 1, 19], making fair comparison difficult. Many publicly available datasets are too small to enable reliable comparison (FUNSD [24], Kleister NDA [45]) or are almost solved, i.e., there is no room for improvement due to annotation errors and near-perfect scores achieved by models nowadays (SROIE [21], CORD [38], RVL-CDIP [17]).

In light of the above circumstances, the review and selection of representative and reliable tasks is of great importance.

3 End-to-end Document Understanding benchmark

The primary motivation for proposing this benchmark was to select datasets covering the broad range of tasks and DU-related problems satisfying the highest quality, difficulty, and licensing criteria.

Importantly, we opt for an end-to-end nature of tasks as opposed to, e.g., problems assuming some prior information on document layout. In particular, there is no structured representation of the underlying text, such as a database-like table given in advance, and it has to be determined from the raw input file as part of the end-to-end process.

We consider the aforementioned principle of end-to-end nature crucial because it ensures measurement to which degree manual workers can be supported in their repetitive tasks, i.e., how the ultimate goal of document understanding systems is supported in real-world applications. The said *alignment with real applications* is a vital characteristic of a good benchmark [29, 43].

3.1 Selected datasets

Extensive documentation of the selection process, including the datasheet, is available in Appendices A-H and in the supplementary materials. Table 1 summarizes the selected tasks described in detail below, whereas Appendix A covers the complete list of considered datasets and reasons we omitted them.

Lack of the classification, layout analysis and figure QA tasks in this selection results from the fact that none of the available sets fulfills the assumed selection criteria.

Table 1: Comparison of selected datasets with their base characteristics, including information regarding whether an input is an entire document (Doc.) or document excerpt (Exc.)

Task	Size (k documents)			Mean samples per document	Type	Metric	Features		Domain
	Train	Dev	Test				Input	Scanned	
DocVQA	10.2	1.3	1.3	3.9	Visual QA	ANLS	}Doc.	+	Business
InfographicsVQA	4.4	0.5	0.6	5.5	Visual QA	ANLS		-	Open
Kleister Charity	1.7	0.4	0.6	7.8	KIE	F1		+/-	Legal
PWC	0.2	0.06	0.12	25.5	KIE	ANLS ²		-	Scientific
DeepForm★	0.7	0.1	0.3	4.8	KIE	F1		+/-	Finances
WikiTableQuestions★	1.4	0.3	0.4	11.3	Table QA	Acc.	}Exc.	-	Open
TabFact★	13.2	1.7	1.7	7.1	Table NLI	Acc.		-	Open

The ★ symbol denotes that the dataset was reformulated or modified to improve its quality or align with the Document Understanding paradigm (see Table 2 and Appendix C). This symbol is not used to distinguish minor changes, such as data deduplication introduced in multiple datasets (Appendix B).

DocVQA. Dataset for Question Answering over single-page excerpts from various real-world industry documents. Typical questions present here might require comprehension of images, free text, tables, lists, forms, or their combination [33]. The best-performing solutions so far make use of layout-aware multi-modal models employing either encoder-decoder or sequence labeling architectures [40, 56].

InfographicsVQA. The task of answering questions about visualized data from a diverse collection of infographics, where the information needed to answer a question may be conveyed by text, plots, graphical or layout elements. Currently, the best result is obtained by an encoder-decoder model [32, 40].

Kleister Charity. A task for extracting information about charity organizations from their published reports is considered, as it is characterized by careful manual annotation by linguists and a significant gap to human performance [45]. It addresses important areas, namely high layout variability (lack of templates), need for performing an OCR, the appearance of long documents, and multiple spatial features (e.g., tables, lists, and titles).

PWC★. Papers with Code Leaderboards dataset was designed to extract result tuples from machine learning papers, including information on task, dataset, metric name, score. The best performing approach involves a multi-step pipeline, with modules trained separately on identified subproblems [26]. In contrast to the original formulation, we provide a complete paper as input instead of the table. This approach allows us to treat the problem as an end-to-end Key Information Extraction task with grouped variables (Appendix C).

DeepForm★. KIE dataset consisting of socially important documents related to election spending. The task is to extract contract number, advertiser name, amount paid, and air dates from advertising disclosure forms submitted to the Federal Communications Commission [47]. We use a subset of distributed datasets and improve annotations errors and make the annotations between subsets for different years consistent (Appendix C).

WikiTableQuestions (WTQ)★. Dataset for QA over semi-structured HTML tables sourced from Wikipedia. The authors intended to provide complex questions, demanding multi-step reasoning on a series of entries in the given table, including comparison and arithmetic operations [39]. The problem is commonly approached assuming a semantic parsing paradigm, with an intermediate state of formal meaning representation, e.g., inferred query or predicted operand to apply on selected cells [58, 18]. We reformulate the task as document QA by rendering the original HTML and restrict available information to layout given by visible lines and token positions (Appendix C).

TabFact★. To study fact verification with semi-structured evidence over relatively clean and simple tables collected from Wikipedia, entailed and refuted statements corresponding to a single row or cell were prepared by the authors of TabFact [7]. Without being affected by the simplicity of binary classification, this task poses challenges due to the complex linguistic and symbolic reasoning

²The ANLS metric used in PWC, representing KIE with property groups, differs from one used in VQA. Since it is not known how many groups are to be returned, the basis of the metric is the F1 score (in contrast to accuracy). Moreover, we require exact math for numerical variables. See implementation in the repository.

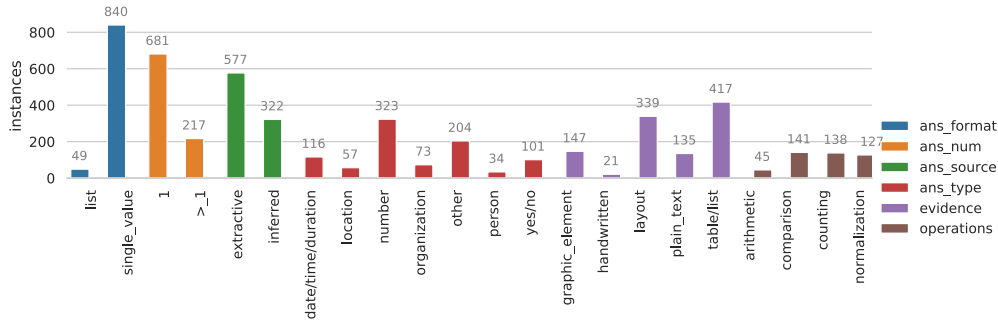


Figure 2: Number of annotated instances in each diagnostic subset category. All datasets in total.

required to perform with high accuracy. Analogously to WTQ, we render tables and reformulate the task as document NLI (Appendix C).

3.2 Diagnostic subsets

As pointed out by Ruder, *to better understand the strengths and weaknesses of our models, we furthermore require more fine-grained evaluation* [43]. We propose several auxiliary validation subsets, spanning across all the tasks, to improve result analysis and aid the community in identifying where to focus its efforts. A detailed description of these categories and related annotation procedures is provided in Appendix F.

Answer characteristic. We consider four features regarding the shallow characteristic of the answer. First, we indicate whether the answer is provided in the text explicitly in exact form (*extractive* data point) or has to be inferred from the document content (*abstractive* one). The second category includes, e.g., all the cases where value requires normalization before being returned (e.g., changing the date format). Next, we distinguish expected answers depending on whether they contain a *single value* or *list* of values. Finally, we decided to recognize several popular data types depending on shapes or class of expected named entity, i.e., to distinguish *date*, *number*, *yes/no*, *organization*, *location*, and *person* classes.

Evidence form. As we intend to analyze systems dealing with rich data, it is natural to study the performance w.r.t. the form that evidence is presented within the analyzed document. We distinguished *table/list*, *plain text*, *graphic element*, *layout*, and *handwritten* categories.

Required operation. Finally, we distinguish whether i.e., *arithmetic operation*, *counting*, *normalization* or some form of *comparison* has to be performed to answer correctly.

Table 2: Brief characteristics of our contribution, major fixes and modifications introduced to particular datasets. The enhancements of "Reformulation as DU" or "Improving data splits" are marked with \star and are sufficient to consider the dataset unique; hence, achieved results are not comparable to the previously reported. See Appendix C for a full description of tasks processing.

Dataset	Diagnostic sets	Unified format	Human performance	Manual annotation	Reformulation as DU	Improved split
DocVQA	+	+	-	-	-	-
InfographicsVQA	+	+	-	-	-	-
Kleister Charity	+	+	-	-	-	-
PWC \star	+	+	+	+	+	+
DeepForm \star	+	+	+	+	-	+
WikiTableQuestions \star	+	+	+	-	+	+
TabFact \star	+	+	-	-	+	-

Datasets included in the benchmark differ in task type, origin, and answer form. As their random samples were annotated, diagnostic categories are not distributed uniformly and reflect the character of the problems encountered in a particular task (see Figures 10–11 in the Appendix). For example, the requirement of answer normalization is prevalent in KIE tasks of DeepForm, PWC, and Kleister Charity but not elsewhere. Consequently, the general framework of diagnostic subsets we designed can be used not only to analyze model performance but also to characterize the datasets themselves.

3.3 Intended use

Data. We propose a unified data format for storing information in the Document Understanding domain and deliver converted datasets as part of the released benchmark (all selected datasets are hosted on the <https://duebenchmark.com/data> and can be downloaded from there). It assumes three interconnected dataset, document annotation and document content levels. The dataset level is intended for storing the general metadata, e.g., name, version, license, and source. The documents annotation level is intended to store annotations available for individual documents within datasets and related metadata (e.g., external identifiers). The content level store information about output and metadata from a particular OCR engine that was used to process documents (Appendix G).

Evaluation protocol. To evaluate a system on the DUE benchmark, one must create a JSON file with the results (in the data format mentioned above) based on the provided test data for each dataset and then upload all of the data to the website. Moreover, we establish a set of rules (Appendix H) which guarantees that all the benchmark submissions will be fair to compare, reproducible, and transparent (e.g., training performed on a development set is not allowed).

Leaderboard. We provide an online platform for the evaluation of Document Understanding models. To keep an objective means of comparison with the previously published results, we decided to retain the initially formulated metrics. To calculate the global score we resort to an arithmetic mean of different metrics due to its simplicity and straightforward calculation.³ In our platform we focus on customization, e.g., multiple leaderboards are available, and it is up to the participant to decide whether to evaluate the model on an entire benchmark or particular category. Moreover, we pay attention to the explanation by providing means to analyze the performance concerning document or problem types (e.g., using the diagnostic sets we provide).⁴

4 Experiments

Following the evaluation protocol, the training is run three times for each configuration of model size, architecture, and OCR engine. We performed OCR pre-processing stage for DocVQA, InfographicsVQA, Kleister Charity, and DeepForm datasets since they have PDF (mix of scans and born-digital documents) or image files as an input. PWC, WikiTableQuestions and TabFact datasets contain all born-digital documents so the ground true data are available and there is no need to run OCR engine (see Appendix C). In both cases, the pre-processing stage as an output return textual tokens and their locations (bounding boxes and page number) as a list (as a result the reading order is also provided).

4.1 Baselines

The focus of the experiments was to calculate baseline performance using a simple and popular model capable of solving all tasks without introducing any task-specific alterations. Employed methods were based on the previously released T5 model with a generic layout-modeling modification and pretraining.

T5. Text-to-text Transformer is particularly useful in studying performance on a variety of sequential tasks. We decided to rely on its extended version to identify the current level of performance on the chosen tasks and to facilitate future research by providing extendable architecture with a straightforward training procedure that can be applied to all of the proposed tasks in an end-to-end manner [41].

³Scores on the DocVQA and InfographicsVQA test sets are calculated using the official website.

⁴We intend to gather datasets not included in the present version of the benchmark to facilitate evaluations in an entire field of DU, regardless of if they are included in the current version of the leaderboard.

Table 3: Best results of particular model configuration in relation to human performance and external best. The external bests marked with — were omitted due to the significant changes in the data sets. *U* stands for unsupervised pretraining.

Dataset / Task type	Score (task-specific metric)					Human
	T5	T5+2D	T5+U	T5+2D+U	External best	
DocVQA	70.4 \pm 2.1	69.8 \pm 0.7	76.3 \pm 0.3	81.0 \pm 0.2	87.1 [40]	98.1
InfographicsVQA	36.7 \pm 0.6	39.2 \pm 1.0	37.1 \pm 0.2	46.1 \pm 0.1	61.2 [40]	98.0
Kleister Charity	74.3 \pm 0.3	72.6 \pm 1.1	76.0 \pm 0.1	75.9 \pm 0.7	83.6 [63]	97.5
PWC [★]	25.3 \pm 3.3	25.7 \pm 1.0	27.6 \pm 0.6	26.8 \pm 1.8	—	69.3
DeepForm [★]	74.4 \pm 0.6	74.0 \pm 0.7	82.9 \pm 0.9	83.3 \pm 0.3	—	98.5
WikiTableQuestions [★]	33.3 \pm 0.7	30.8 \pm 1.9	38.1 \pm 0.1	43.3 \pm 0.4	—	76.7
TabFact [★]	58.9 \pm 0.5	58.0 \pm 0.3	76.0 \pm 0.1	78.6 \pm 0.1	—	92.1
Visual QA	53.6	54.5	56.7	63.5	n/a	98.1
KIE	69.1	67.7	74.8	76.4	n/a	88.4
Table QA/NLI	29.4	29.0	38.0	39.3	n/a	84.4
Overall	50.7	50.4	56.5	59.8	n/a	90.3

T5+2D. Extension of the model we propose assumes the introduction of 2D positional bias that has been shown to perform well on tasks that demand layout understanding [56, 40, 63]. We rely on 2D bias in a form introduced in TILT model [40] and provide its first open-source implementation (available in supplementary materials). We expect that comprehension of spatial relationships achieved in this way will be sufficient to demonstrate that methods from the plain-text NLP can be easily outperformed in the DUE benchmark.

Unsupervised pretraining. We constructed a corpus of documents with a visually rich structure, based on 480k PDF files from the UCSF Industry Documents Library. It is used with a T5-like masked language model pretraining objective but in a salient span masking scheme where named entities are preferred over random tokens [41, 15]. An expected gain from its use is to tune 2D biases and become more robust to OCR errors and incorrect reading order.⁵

Human performance. We relied on the original estimation for DocVQA, InfographicsVQA, Charity, and TabFact datasets. For the PWC, WTQ and DeepForm estimation of human performance, we used the help of professional in-house annotators who are full-time employees of our company (see Appendix E). Each dataset was handled by two annotators; the average of their scores, when validated against the gold standard, is treated as the human performance (see Table 3). Interestingly, human scores on PWC are relatively low in terms of ANLS value – we explained this and justified keeping the task in Appendix C.

4.2 Results

Comparison of the best-performing baselines to human performance and top results reported in the literature is presented in Table 3. In several cases, there is a small difference between the performance of our baselines and the external best. It can be attributed to several factors. First, the best results previously obtained on the tasks were task-specific, i.e., were explicitly designed for a particular task and did not support processing other datasets within the benchmark. Secondly, there are differences between the evaluation protocol that we assume and what the previous authors assumed (e.g., we do not allow training models on the development sets, we require reporting an average of multiple runs, we disallow pretraining on datasets that might lead to information leak). Thirdly, our baseline could not address examples demanding vision comprehension as it does not process image inputs. Finally, there is the case of Kleister Charity. An encoder-decoder model we relied on as a one-to-fit-all baseline cannot process an entire document due to memory limitations. As a result, the score was lower as we consumed only a part of the document. Note that external bests for reformulated tasks are no longer applicable to the benchmark in its present, more demanding form.

⁵Details of the training procedure, such as used hyperparameters and source code, are available in the repository accompanying the paper.

Irrespective of the task and whether our competitive baselines or external results are considered, there is still a large gap to humans, which is desired for novel baselines. Moreover, one can notice that the addition of 2D positional bias to the T5 architecture leads to better scores assuming the prior pretraining step, which is yet another result we anticipated as it suggests that considered tasks have an essential component of layout comprehension.

Interestingly, the performance of the model can be significantly enhanced (up to 20.6 points difference for TabFact dataset and T5+2D+U model) by providing additional data for the said unsupervised pretraining. Thus, the results not only support the premise that understanding 2D features demand more unlabeled data than the chosen datasets can offer but also lay a common ground between them, as the same layout-specific pretraining improved performance on all of them independently. This observation confirms that the notion of layout is a vital part of the chosen datasets.

4.3 Challenges of the Document Understanding domain

Owing to its end-to-end nature and heterogeneity, Document Understanding is the touchstone of Machine Learning. However, the challenges begin to pile up due to the mere form a document is available in, as there is a widespread presence of analog materials such as scanned paper records. In the analysis below, we aim to explore the field of DU from the perspective of the model’s development and point out the most critical limiting factors for achieving satisfying results.

Impact of OCR quality. We present detailed results for Azure CV and Tesseract OCR engine in Table 5. The differences in scores are huge for most of the datasets (up to 18.4% in DocVQA) with a clean advantage for Azure CV. Consequently, we see that architectures evaluated with different OCR engines are incomparable, e.g., the choice of an OCR engine may impact results more than the choice of model architecture. Moreover, with the usage of our diagnostic datasets we can observe that Tesseract struggle the most with *Handwritten* and *Table/list* categories in comparison to *Plain text* category. It is worth noting that we see a bigger difference in the results between Azure CV and Tesseract for *Extractive* category, which suggest that we should use better OCR engines especially for that kind of problems.

Requirement of multi-modal comprehension. In addition to layout and textual semantics, part of the covered problems demand a Computer Vision component, e.g., to detect a logo, analyze a figure, recognize text style, determine whether the document was signed or the checkbox nearby was selected. This has been confirmed by ablation studies performed by Powalski et al. [40] for the DocVQA and by the fact that models with vision component achieve better performance on leaderboards for datasets such as DocVQA and the InfographicsVQA datasets [40, 56, 23, 22]. Thus, Document Understanding naturally incorporates challenges of both multi-modality and each modality individually (but not for all tasks equally, see Figures 10–11 in the Appendix). Since none of our baselines contain a vision component, we underperform on the category of problems requiring multi-modality, as is visible on the diagnostic dataset we proposed. Nevertheless, better performance of the T5+2D model suggests that part of the problems considered as *visual*, can be to some extent approximated by solely using the words’ spatial relationships (e.g., text curved around a circle, located in the top-left corner of the page presumably has the logo inside).

Single architecture for all datasets. It is common that token-level annotation is not available, and one receives merely key-value or question-answer pairs assigned to the document. Even in problems of extractive nature, token spans cannot be easily obtained, and consequently, the application of state-of-the-art architectures from other tasks is not straightforward. In particular, authors attempting Document Understanding problems in sequence labeling paradigms were forced to rely on faulty handcrafted heuristics [40]. In the case of our baseline models, this problem is addressed straightforwardly by assuming a sequence-to-sequence paradigm that does not make use of token-level annotation. This solution, however, comes with a trade-off of low performance on datasets requiring comprehension of long documents, such as Kleister Charity (see Table 4).

Table 4: F1 score on the Kleister Charity challenge with various maximum input sequence lengths.

Dataset	Maximum input sequence length			
	1024	2048	4096	6144 (max)
Kleister Charity	56.6	66	73.2	75.9

Table 5: Scores for different OCR engines and datasets with T5+2D model performing on 1024 tokens.

OCR	DocVQA	IVQA	Charity	DeepForm	Average	Average scores for different diagnostic categories				
						Extractive	Inferred	Handwritten	Table/list	Plain text
Azure CV (v3.2)	71.8	40.0	57.7	74.8	61.1	51.3	33.0	31.3	46.0	65.3
Tesseract (v4.0)	55.7	28.3	55.7	66.8	51.6	43.1	29.5	12.5	27.2	61.1

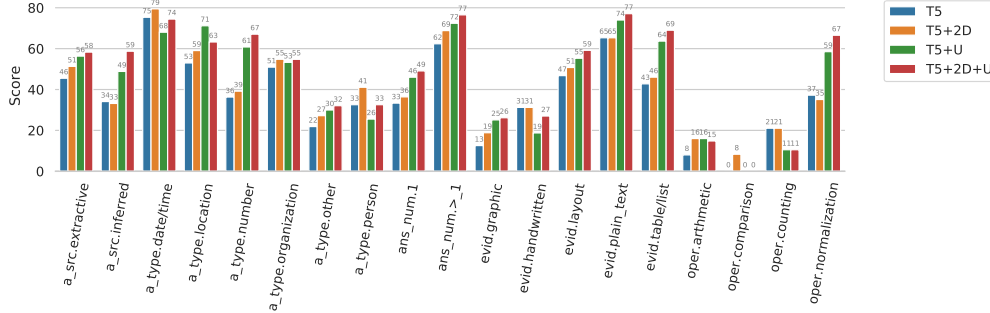


Figure 3: Results for diagnostic subsets. See Appendix F for detailed description of these categories.

Diagnostic dataset. Our diagnostic datasets are an important part of the analysis of different challenges in general (e.g., OCR quality or multi-modal comprehension, as we mentioned above) and for debugging different types of architectural decisions (see Figure 3). For example, we can observe a big advantage of unsupervised pretraining in the *inferred*, *number*, *table/list* categories, which shows the importance of a good dataset for specific problems (dataset used for pretraining the original T5 model has a small number of documents containing tables). The most problematic categories for all models were those related to complex logic operations: *arithmetic*, *counting*, *comparison*.

5 Conclusions

To efficiently pass information to the reader, writers often assume that structured forms such as tables, graphs, or infographics are more accessible than sequential text due to human visual perception and our ability to understand a text’s spatial surroundings. We investigate the problem of correctly measuring the progress of models able to comprehend such complex documents and propose a benchmark – a suite of tasks that balance factors such as quality of a document, importance of layout information, type and source of documents, task goal, and the potential usability in modern applications.

We aim to track the future progress on them with the website prepared for transparent verification and analysis of the results. The former is facilitated by the diagnostics subsets we derived to measure vital features of the Document Understanding systems. Finally, we provide a set of solid baselines, datasets in the unified format, and released source code to bootstrap the research on the topic.

Acknowledgments and disclosure of funding

The authors would like to thank Samuel Bowman, Łukasz Garncarek, Dimosthenis Karatzas, Minesh Mathew, Zofia Prochoroff, and Rubèn Pérez Tito for the helpful discussions on the draft of the paper. Moreover, we thank the reviewers of both rounds of the NeurIPS 2021 Datasets and Benchmarks Track for their comments and suggestions that helped improve the paper.

The Smart Growth Operational Programme supported this research under project no. POIR.01.01.01-00-0877/19-00 (*A universal platform for robotic automation of processes requiring text comprehension, with a unique level of implementation and service automation*).

References

- [1] S. Appalaraju, B. Jasani, B. U. Kota, Y. Xie, and R. Manmatha. DocFormer: End-to-end transformer for document understanding, 2021.
- [2] T. Bayer, J. Franke, U. Kressel, E. Mandler, M. Oberländer, and J. Schürmann. Towards the understanding of printed documents. In H. S. Baird, H. Bunke, and K. Yamamoto, editors, *Structured Document Image Analysis*, pages 3–35. Springer Berlin Heidelberg, Berlin, Heidelberg, 1992.
- [3] I. Chalkidis, E. Fergadiotis, P. Malakasiotis, and I. Androutsopoulos. Large-scale multi-label text classification on EU legislation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6314–6322, Florence, Italy, July 2019. Association for Computational Linguistics.
- [4] R. Chaudhry, S. Shekhar, U. Gupta, P. Maneriker, P. Bansal, and A. Joshi. Leaf-qa: Locate, encode attend for figure question answering. In *2020 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 3501–3510, 2020.
- [5] L. Chen, X. Chen, Z. Zhao, D. Zhang, J. Ji, A. Luo, Y. Xiong, and K. Yu. WebSRC: A dataset for web-based structural reading comprehension, 2021.
- [6] W. Chen, M. Chang, E. Schlinger, W. Wang, and W. Cohen. Open question answering over tables and text. *Proceedings of ICLR 2021*, 2021.
- [7] W. Chen, H. Wang, J. Chen, Y. Zhang, H. Wang, S. Li, X. Zhou, and W. Y. Wang. TabFact : A large-scale dataset for table-based fact verification. In *International Conference on Learning Representations (ICLR)*, Addis Ababa, Ethiopia, April 2020.
- [8] W. Chen, H. Zha, Z. Chen, W. Xiong, H. Wang, and W. Wang. HybridQA: A dataset of multi-hop question answering over tabular and textual data, 2021.
- [9] M. Cho, R. K. Amplayo, S. won Hwang, and J. Park. Adversarial TableQA: Attention supervision for question answering on tables, 2018.
- [10] M. Dehghani. Toward document understanding for information retrieval. *SIGIR Forum*, 51(3):27–31, Feb. 2018.
- [11] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*, pages 248–255, 2009.
- [12] T. I. Denk and C. Reisswig. BERTgrid: Contextualized Embedding for 2D Document Representation and Understanding. In *Workshop on Document Intelligence at NeurIPS 2019*, 2019.
- [13] F. Esposito, D. Malerba, G. Semeraro, and S. Ferilli. Knowledge revision for document understanding. In *ISMIS*, 1997.
- [14] V. Gupta, M. Mehta, P. Nokhiz, and V. Srikumar. INFOTABS: Inference on tables as semi-structured data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 2309–2324, Online, July 2020. Association for Computational Linguistics.
- [15] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang. Retrieval augmented language model pre-training. In *ICML*, 2020.
- [16] R. M. Haralick. Document image understanding: Geometric and logical layout. In *CVPR*, volume 94, pages 385–390, 1994.
- [17] A. W. Harley, A. Ufkes, and K. G. Derpanis. Evaluation of deep convolutional nets for document image classification and retrieval. In *International Conference on Document Analysis and Recognition (ICDAR)*, 2015.

- [18] J. Herzig, P. K. Nowak, T. Müller, F. Piccinno, and J. Eisenschlos. TaPas: Weakly supervised table parsing via pre-training. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 4320–4333, Online, July 2020. Association for Computational Linguistics.
- [19] T. Hong, D. Kim, M. Ji, W. Hwang, D. Nam, and S. Park. BROS: A layout-aware pre-trained language model for understanding documents, 2021.
- [20] Z. Huang, K. Chen, J. He, X. Bai, D. Karatzas, S. Lu, and C. Jawahar. ICDAR2019 competition on scanned receipt OCR and information extraction. In *ICDAR*, 2019.
- [21] Z. Huang, K. Chen, J. He, X. Bai, D. Karatzas, S. Lu, and C. V. Jawahar. Icdar2019 competition on scanned receipt ocr and information extraction. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1516–1520, 2019.
- [22] ICDAR. Leaderboard of the Document Visual Question Answering - Infographics VQA. <https://rrc.cvc.uab.es/?ch=17&com=evaluation&task=3> (accessed September 30, 2021), 2021.
- [23] ICDAR. Leaderboard of the Document Visual Question Answering - Single Document VQA. <https://rrc.cvc.uab.es/?ch=17&com=evaluation&task=1> (accessed September 30, 2021), 2021.
- [24] G. Jaume, H. K. Ekenel, and J.-P. Thiran. FUNSD: A dataset for form understanding in noisy scanned documents, 2019.
- [25] K. V. Jobin, A. Mondal, and C. V. Jawahar. DocFigure: A dataset for scientific document figure classification. In *2019 International Conference on Document Analysis and Recognition Workshops (ICDARW)*, volume 1, pages 74–79, 2019.
- [26] M. Kardas, P. Czapla, P. Stenetorp, S. Ruder, S. Riedel, R. Taylor, and R. Stojnic. AxCell: Automatic extraction of results from machine learning papers, 2020.
- [27] A. R. Katti, C. Reisswig, C. Guder, S. Brarda, S. Bickel, J. Höhne, and J. B. Faddoul. Chargrid: Towards Understanding 2D Documents. *ArXiv*, abs/1809.08799, 2018.
- [28] A. Kembhavi, M. Seo, D. Schwenk, J. Choi, A. Farhadi, and H. Hajishirzi. Are you smarter than a sixth grader? Textbook question answering for multimodal machine comprehension. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5376–5384, 2017.
- [29] S. Kounev, K. Lange, and J. von Kistowski. *Systems Benchmarking: For Scientists and Engineers*. Springer International Publishing, 2020.
- [30] M. Li, Y. Xu, L. Cui, S. Huang, F. Wei, Z. Li, and M. Zhou. DocBank: A benchmark dataset for document layout analysis, 2020.
- [31] B. P. Majumder, N. Potti, S. Tata, J. B. Wendt, Q. Zhao, and M. Najork. Representation learning for information extraction from form-like documents. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6495–6504, Online, July 2020. Association for Computational Linguistics.
- [32] M. Mathew, V. Bagal, R. P. Tito, D. Karatzas, E. Valveny, and C. V. Jawahar. InfographicVQA, 2021.
- [33] M. Mathew, D. Karatzas, and C. Jawahar. DocVQA: A dataset for VQA on document images. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV)*, pages 2200–2209, January 2021.
- [34] N. Methani, P. Ganguly, M. M. Khapra, and P. Kumar. PlotQA: Reasoning over scientific plots. In *The IEEE Winter Conference on Applications of Computer Vision (WACV)*, March 2020.
- [35] L. Nan, C. Hsieh, Z. Mao, X. V. Lin, N. Verma, R. Zhang, W. Kryściński, N. Schoelkopf, R. Kong, X. Tang, M. Mutuma, B. Rosand, I. Trindade, R. Bandaru, J. Cunningham, C. Xiong, and D. Radev. FeTaQA: Free-form table question answering, 2021.

- [36] D. Niyogi and S. N. Srihari. A rule-based system for document understanding. In *Proceedings of the Fifth AAAI National Conference on Artificial Intelligence*, pages 789–793, 1986.
- [37] R. B. Palm, F. Laws, and O. Winther. Attend, copy, parse end-to-end information extraction from documents. *International Conference on Document Analysis and Recognition (ICDAR)*, 2019.
- [38] S. Park, S. Shin, B. Lee, J. Lee, J. Surh, M. Seo, and H. Lee. CORD: A consolidated receipt dataset for post-ocr parsing. In *Document Intelligence Workshop at NeurIPS*, 2019.
- [39] P. Pasupat and P. Liang. Compositional semantic parsing on semi-structured tables. *CoRR*, abs/1508.00305, 2015.
- [40] R. Powalski, Ł. Borchmann, D. Jurkiewicz, T. Dwojak, M. Pietruszka, and G. Pałka. Going full-tilt boogie on document understanding with text-image-layout transformer. In J. Lladós, D. Lopresti, and S. Uchida, editors, *Document Analysis and Recognition – ICDAR 2021*, pages 732–747, Cham, 2021. Springer International Publishing.
- [41] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text Transformer. *Journal of Machine Learning Research*, 21(140):1–67, 2020.
- [42] M. Rimol. Gartner Forecasts Worldwide Hyperautomation-Enabling Software Market to Reach Nearly \$600 Billion by 2022. <https://www.gartner.com/en/newsroom/press-releases/2021-04-28-gartner-forecasts-worldwide-hyperautomation-enabling-software-market-to-reach-nearly-600-billion-by-2022>, 2021.
- [43] S. Ruder. Challenges and Opportunities in NLP Benchmarking. <http://ruder.io/nlp-benchmarking>, 2021.
- [44] Z. Shen, K. Lo, L. L. Wang, B. Kuehl, D. S. Weld, and D. Downey. Incorporating visual layout structures for scientific text classification, 2021.
- [45] T. Stanisławek, F. Galiński, A. Wróblewska, D. Lipiński, A. Kaliska, P. Rosalska, B. Topolski, and P. Biecek. Kleister: Key information extraction datasets involving long documents with complex layouts, 2021.
- [46] H. Sun, Z. Kuang, X. Yue, C. Lin, and W. Zhang. Spatial dual-modality graph reasoning for key information extraction, 2021.
- [47] S. Svetlichnaya. DeepForm: Understand structured documents at scale. https://wandb.ai/stacey/deepform_v1/reports/DeepForm-Understand-Structured-Documents-at-Scale--Vml1dzoyODQ3Njg, 2020.
- [48] A. Talmor, O. Yoran, A. Catav, D. Lahav, Y. Wang, A. Asai, G. Ilharco, H. Hajishirzi, and J. Berant. Multimodalqa: Complex question answering over text, tables and images. *CoRR*, abs/2104.06039, 2021.
- [49] R. Tanaka, K. Nishida, and S. Yoshida. VisualMRC: Machine reading comprehension on document images. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(15):13878–13888, May 2021.
- [50] S. L. Taylor, D. Dahl, M. Lipshutz, C. Weir, L. M. Norton, R. Nilson, and M. Linebarger. Integrated text and image understanding for document understanding. In *HLT*, 1994.
- [51] H. M. Vu and D. T. Nguyen. Revising FUNSD dataset for key-value detection in document images. *CoRR*, abs/2010.05322, 2020.
- [52] A. Wang, Y. Pruksachatkun, N. Nangia, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. Superglue: A stickier benchmark for general-purpose language understanding systems. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

- [53] A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. Bowman. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium, Nov. 2018. Association for Computational Linguistics.
- [54] T.-L. Wu, S. Singh, S. Paul, G. Burns, and N. Peng. MELINDA: A multimodal dataset for biomedical experiment method classification. *ArXiv*, abs/2012.09216, 2020.
- [55] Y. Xu, M. Li, L. Cui, S. Huang, F. Wei, and M. Zhou. LayoutLM: Pre-training of text and layout for document image understanding, 2019.
- [56] Y. Xu, Y. Xu, T. Lv, L. Cui, F. Wei, G. Wang, Y. Lu, D. Florencio, C. Zhang, W. Che, M. Zhang, and L. Zhou. LayoutLMv2: Multi-modal pre-training for visually-rich document understanding, 2020.
- [57] S. Yacoub. Automated quality assurance for document understanding systems. *IEEE Software*, 20(3):76–82, 2003.
- [58] P. Yin, G. Neubig, W.-t. Yih, and S. Riedel. TaBERT: Pretraining for joint understanding of textual and tabular data. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8413–8426, Online, July 2020. Association for Computational Linguistics.
- [59] X. Zhai, J. Puigcerver, A. Kolesnikov, P. Ruysen, C. Riquelme, M. Lucic, J. Djolonga, A. S. Pinto, M. Neumann, A. Dosovitskiy, L. Beyer, O. Bachem, M. Tschannen, M. Michalski, O. Bousquet, S. Gelly, and N. Houlsby. A large-scale study of representation learning with the visual task adaptation benchmark, 2020.
- [60] X. Zheng, D. Burdick, L. Popa, and N. X. R. Wang. Global table extractor (gte): A framework for joint table identification and cell structure recognition using visual context. *2021 IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 697–706, 2021.
- [61] X. Zhong, J. Tang, and A. J. Yepes. PubLayNet: largest dataset ever for document layout analysis. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1015–1022. IEEE, Sep. 2019.
- [62] F. Zhu, W. Lei, Y. Huang, C. Wang, S. Zhang, J. Lv, F. Feng, and T.-S. Chua. TAT-QA: A question answering benchmark on a hybrid of tabular and textual content in finance. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 3277–3287, Online, Aug. 2021. Association for Computational Linguistics.
- [63] Łukasz Garncarek, R. Powalski, T. Stanisławek, B. Topolski, P. Halama, and F. Graliński. LAMBERT: Layout-aware (language) modeling using bert for information extraction, 2020.

A Considered datasets

A.1 Desired characteristics

End-to-end nature. As the value and importance of Document Understanding result from its application to process automation, a good benchmark should measure to which degree workers can be supported in their tasks. Though Layout Analysis is oldest of the Document Understanding problems, its output is often not an end in itself but rather a half-measure disconnected from the final information the system is used for. We also remove all tasks which as an input takes collection of documents.

Quality. Availability of high-quality annotation was a condition *sine qua non* for a task to qualify. To ensure the highest annotation quality, we excluded resources prepared using a distant annotation procedure, e.g., classification tasks where entire sources were labeled instead of individual instances, or templated question-answer pairs.

Difficulty. As it makes no sense to measure progress on solved problems, only tasks with a substantial gap between human performance and state-of-the-art models were considered. In the case of promising tasks lacking a human baseline, we provided our estimation. Moreover, we remove all tasks where free text was dominated in documents (we don't need to use layout or visual features).

Licensing. In publishing our benchmark, we are making efforts to ensure the highest standards for the future of the machine learning community. Only tasks with a permissive license to use annotations and data for further research can be considered.

At the same time, we recognized it is essential to approach the benchmark construction holistically, i.e., to carefully select tasks from diverse domains and types in the rare cases where datasets are abundant.

A.2 Datasets selection process

The review protocol consisted of a manual search in specific databases, repositories and distribution services. The scientific resources included in the search were:

- <https://paperswithcode.com/datasets/>
- <https://datasetsearch.research.google.com/>
- <https://data.mendeley.com/>
- <https://arxiv.org/search/>
- <https://github.com/>
- <https://allenai.org/data/>
- <https://www.semanticscholar.org/>
- <https://scholar.google.com/>
- <https://academic.microsoft.com/home>

Results were reviewed by one of authors of the present paper and the resources related to classification, KIE, QA, MRC, and NLI over complex documents, figures, and tables were identified as potentially relevant (in accordance with inclusion criteria described in Section A.1).

The initial search assumed use of the following keywords: *Question Answering*, *Visual Question Answering*, *Document Question Answering*, *Document Classification*, *Document Dataset*, *Information Extraction*. Additionally, we used *Machine Reading Comprehension*, *Question Answering*, *VQA* in combination with *Document*, and *Visual*, *Document*, *Table*, *Figure*, *Plot*, *Chart*, *Hybrid* in combination with *Question Answering* or *Information Extraction*.

Table 6 presents list of relevant datasets and results of their assessment according to the criteria of end-to-end nature, quality, difficulty, and licensing. Candidate tasks resulted from an extensive review of both literature and data science challenges without accompanying publication and their basic characteristics.

Table 6: Comparison of selected and considered datasets with their base characteristic, including information regarding whether an input is a collection of documents (Col.), entire document (Doc.) or document excerpt (Exc.).

Dataset	Type	Size (thousands)			Selection criteria			Input	Domain	Comment
		Train	Dev	Test	End-to-end	Quality	Difficulty			
Kleister Charity [45]	KIE	1.73	.44	.61	+	+	+	+	Doc.	Finances
PWC [26]	KIE	.2	.06	.12	+	+	+	+	Doc.	Scientific
DeepForm [47]	KIE	.7	.1	.3	+	+	+	+	Doc.	Finances
DocVQA [33]	Visual QA	10.2	1.3	1.3	+	+	+	+	Doc.	Business
InfographicsVQA [32]	Visual QA	4.4	.5	.6	+	+	+	+	Doc.	Open
TabFact [7]	Table NLI	13.2	1.7	1.7	+	+	+	+	Exc.	Open
WTQ [39]	Table QA	1.4	.3	.4	+	+	+	+	Exc.	Open
Kleister NDA [45]	KIE	.25	.08	.2	+	+	-	+	Doc.	Legal
SROIE [20]	KIE	.63	-	.35	+	+	-	+	Doc.	Finances
CORD [38]	KIE	.8	.1	.1	+	+	-	+	Doc.	Finances
Wildreceipt [46]	KIE	1.27	-	.47	+	+	-	+	Doc.	Finances
WebSRC [5]	KIE	4.55	.9	1.0	+	-	+	+	Doc.	Open
FUNSD [24]	KIE	.15	-	.05	+	-	+	+	Doc.	Finances
DocVQA [32]	Visual QA	4.4	.5	.6	-	+	+	+	Col.	Open
TextbookQA [28]	Visual QA	.67	.2	.21	+	-	+	+	Doc.	Educational
MultiModalQA [48]	Visual QA	23.82	2.44	3.66	+	-	+	+	Doc.	Open
VisualMRC [49]	Visual MRC	7	1	2	+	+	-	+	Doc.	Open
RVL-CDIP [17]	Classification	320	40	40	+	+	-	+	Doc.	Finances
DocFigure [25]	Classification	19.8	-	13.1	+	+	-	+	Doc.	Scientific
EURLEX57K [3]	Classification	45	6	6	+	+	-	+	Doc.	Legal
MELINDA [54]	Classification	4.34	.45	.58	+	-	+	+	Doc.	Scientific
S2-VL [44]	DLA	1.3	-	-	-	+	+	+	Doc.	Scientific
DocBank [30]	DLA	398	50	50	-	-	+	+	Doc.	Scientific
Publaynet [61]	DLA	340.4	11.9	12	-	-	+	+	Doc.	Scientific
FinTabNet [60]	DLA	61.8	7.19	7.01	-	+	+	+	Doc.	Finances
PlotQA [34]	Figure QA	157	33.7	33.7	+	-	+	+	Exc.	Open
Leaf-QA [4]	Figure QA	200	40	8.15	+	-	+	+	Exc.	Open
TAT-QA [62]	Table QA	2.2	.28	.28	+	-	+	+	Exc.	Finances
WikiOPS [9]	Table QA	17.28	2.47	4.67	+	+	-	+	Exc.	Open
FeTaQA [35]	Table QA	7.33	1.0	2.0	+	-	+	+	Exc.	Open
HybridQA [8]	Table QA	62.68	3.47	3.46	-	+	+	+	Col.	Open
OTT-QA [6]	Table QA	41.46	2.24	2.16	-	+	+	+	Col.	Open
INFOTABS [14]	Table NLI	1.74	.2	.6	+	+	+	+	Col.	Open

B Minor dataset modifications

Deduplication. Through the systematic analysis and validation of the chosen datasets, we noticed one of the commonly appearing defects is the presence of duplicated annotations. We decided to remove these duplicates from InfographicsVQA (14 annotations from train, two from the dev set), DocVQA (four from train and test sets each), TabFact (309 from train, 53 from dev, and 52 the test set), and WikiTableQuestions (one annotation from each train and test sets).

C Tasks processing and reformulation

Since part of the datasets were reformulated or modified to improve the benchmark quality or align the task with the Document Understanding paradigm, we describe the introduced changes in detail below.

WikiTableQuestions★. We prepare input documents by rendering table-related HTML distributed by authors in *wkhtmltopdf* and crop the resulting files with *pdfcrop*. As these code excerpts do not contain *head* tag with JavaScript and stylesheet references, we use the header from the present version of the Wikipedia website.

Approximately 10% of tables contained at least one *img* tag with a source that is no longer reachable. It results in a question mark icon displayed instead of the image and does not impact the evaluation procedure since the questions here do not require image comprehension.

The original WTQ dataset consists of *training*, *pristine-seen-tables*, and *pristine-unseen-tables* subsets. We treat *pristine-unseen-tables* as a test set and create new training and development sets by rearranging data from *training* and *pristine-seen-tables*. The latter operation is dictated by the leakage of documents in the original formulation, i.e., we consider it undesirable for a document to appear in different splits, even if the question differs. The resulting dataset consists of approximately

Year	Venue	Winners	Runner-up	3rd place
2005	 Pardubice	 Poland (41 pts)	 Sweden (35 pts)	 Denmark (24 pts)
2006	 Rybnik	 Poland (41 pts)	 Sweden (27 pts)	 Denmark (26 pts)
2007	 Abensberg	 Poland (40 pts)	 Great Britain (36 pts)	 Czech Republic (30 pts)
2008	 Holsted	 Poland (40 pts)	 Denmark (39 pts)	 Sweden (38 pts)
2009	 Gorzów Wilk.	 Poland (57 pts)	 Denmark (45 pts)	 Sweden (32 pts)
2010	 Rye House	 Denmark (51 pts)	 Sweden (37 pts)	 Poland (35 pts)
2011	 Balakovo	 Russia (61 pts)	 Denmark (31 pts)	 Ukraine (29+3 pts)
2012	 Gniezno	 Poland (61 pts)	 Australia (44 pts)	 Sweden (26 pts)
Year	Venue	Winners	Runner-up	3rd place

Figure 4: Document in WikiTableQuestions reformulated as Document Understanding.

(Question) After their first place win in 2009, how did Poland place the next year at the speedway junior world championship? (Answer) 3rd place

2100 documents divided in the proportion of 65%, 15%, 20% into training, development, and test sets.

We rely on the original WTQ metric which is a form of Accuracy with normalization (see Pasupat et al. [39] and accompanying implementation).

TabFact★. As the authors of TabFact distribute only CSV files, we resorted to HTML from the WikiTables dump their CSV were presumably generated from.⁶ As Chen et al. [7] dropped some of the columns present in used WikiTable tables, we remove them too, to ensure compatibility with the original TabFact. Rendered files are used analogously to the case of WTQ.

	Nation	v t e Games				Points			Table points
		Played	Won	Drawn	Lost	For	Against	Difference	
1	VVA-Podmoskovye Monino	10	9	0	1	374	119	+255	37
2	Krasny Yar Krasnoyarsk	10	6	0	4	198	255	-57	28
3	Slava Moscow	10	5	1	4	211	226	-15	26
4	Yenisey-STM Krasnoyarsk	10	5	0	5	257	158	+99	25
5	RC Novokuznetsk	10	4	1	5	168	194	-26	23
6	Imperia-Dynamo Penza	10	0	0	10	138	395	-257	10

Figure 5: Document in TabFact reformulated as Document Understanding.

(Claim) To calculate table point, a win be worth 3, a tie be worth 1 and a loss be worth 0

Results differ from TabFact in several aspects, i.e., text in our variant is not normalized, it includes the original formatting, and the tables are more complex due to restoring the original cell merges. All mentioned differences are desired, as we intended to consider raw, unprocessed files without any heuristics or normalization applied.

Another difference we noticed is that tables in the original TabFact are sometimes one row shorter, i.e., they do not contain the last row present in the WikiTable dump. As it should not impact expected answers, we decided to maintain the fidelity to Wikipedia and use the complete table.

We use the original splits into training, development, and test sets and the original Accuracy metric.

DeepForm★. The original DeepForm dataset consists of 2012, 2014, and 2020 subsets differing in terms of annotation quality and documents' diversity. We decided to use only the 2020 subset as for 2014, and 2020 annotations were prepared either automatically or by volunteers, leading to questionable quality. The selected subset was randomly divided into training, development and test set.

We noticed several inconsistencies during the initial analysis that lead us to the manual correction of autodetected: (1) invalid date format; (2) flight start dates earlier than flight end; (3) documents lacking one or more data points.

In addition to the improved 2020 subset, we manually annotated one hundred 2012 documents, as they can pose different challenges (contain different document templates, handwriting, have lower

⁶<http://websail-fe.cs.northwestern.edu/TabEL/tables.json.gz>

COXREPS		REP BUYLINES		Page 1												
Alt: 102200-00 Est ID: May09@2016		Alt: 102200-00 Est ID: May09@2016		Alt: 102200-00 Est ID: May09@2016												
# Client: 2110 # Order: 2110 # Contract: 2110		# Client: 2110 # Order: 2110 # Contract: 2110		# Client: 2110 # Order: 2110 # Contract: 2110												
# Agency: 2110 # Product: 2110 # Order: 2110		# Agency: 2110 # Product: 2110 # Order: 2110		# Agency: 2110 # Product: 2110 # Order: 2110												
Row Code	Buy Line	Day/Time	Length	Rate	Starting Date	Ending Date	# of Wks	Split	Total Spots	Total Dollars	Program Name	Rating	Imprmt Pct	Rate	Last Activity	Last Mod/Rev
	#CASH #FSMRT															
1		Tue/5-6A	30S	\$10	May12/20	May12/20	1	1	1	\$10	NEWS10 GOOD MORN -SA	0.9	2.1	0.9	May04/20	Rev #0: A
Contract Comment: NEWS10 GOOD MORN -SA																
2		Wed/5-6A	30S	\$10	May13/20	May13/20	1	1	1	\$10	NEWS10 GOOD MORN -SA	0.9	2.1	0.9	May04/20	Rev #0: A
Contract Comment: NEWS10 GOOD MORN -SA																
3		Thu/5-6A	30S	\$10	May14/20	May14/20	1	1	1	\$10	NEWS10 GOOD MORN -SA	0.9	2.1	0.9	May04/20	Rev #0: A
Contract Comment: NEWS10 GOOD MORN -SA																
4		Mon/5-6A	30S	\$10	May18/20	May18/20	1	1	1	\$10	NEWS10 GOOD MORN -SA	0.9	2.1	0.9	May04/20	Rev #0: A
Contract Comment: NEWS10 GOOD MORN -SA																
5		Wed/6-7A	30S	\$15	May13/20	May13/20	1	1	1	\$15	NEWS10 GOOD MORN -6A	2.2	5.3	2.2	May04/20	Rev #0: A
Contract Comment: NEWS10 GOOD MORN -6A																
6		Thu/6-7A	30S	\$15	May14/20	May14/20	1	1	1	\$15	NEWS10 GOOD MORN -6A	2.2	5.3	2.2	May04/20	Rev #0: A
Contract Comment: NEWS10 GOOD MORN -6A																
7		Fri/6-7A	30S	\$15	May15/20	May15/20	1	1	1	\$15	NEWS10 GOOD MORN -6A	2.2	5.3	2.2	May04/20	Rev #0: A
Contract Comment: NEWS10 GOOD MORN -6A																
8		Mon/6-7A	30S	\$15	May18/20	May18/20	1	1	1	\$15	NEWS10 GOOD MORN -6A	2.2	5.3	2.2	May04/20	Rev #0: A
Contract Comment: NEWS10 GOOD MORN -6A																
9		Tue/7-8A	30S	\$20	May12/20	May12/20	1	1	1	\$20	CBS THIS MORNING	3.0	7.3	3.0	May04/20	Rev #0: A
Contract Comment: CBS THIS MORNING																
10		Thu/7-8A	30S	\$20	May14/20	May14/20	1	1	1	\$20	CBS THIS MORNING	3.0	7.3	3.0	May04/20	Rev #0: A
Contract Comment: CBS THIS MORNING																
11		Mon/7-8A	30S	\$20	May18/20	May18/20	1	1	1	\$20	CBS THIS MORNING	3.0	7.3	3.0	May04/20	Rev #0: A
Contract Comment: CBS THIS MORNING																
12		Tue/9-10A	30S	\$10	May12/20	May12/20	1	1	1	\$10	FAMILY FEUD/ AMERICA SAYS	2.0	4.8	2.0	May04/20	Rev #2: NZ
Contract Comment: FAMILY FEUD/ AMERICA SAYS																
13		Thu/9-10A	30S	\$10	May14/20	May14/20	1	1	1	\$10	FAMILY FEUD/ AMERICA SAYS	2.0	4.8	2.0	May04/20	Rev #2: NZ
Contract Comment: FAMILY FEUD/ AMERICA SAYS																
14		Fri/9-10A	30S	\$10	May15/20	May15/20	1	1	1	\$10	FAMILY FEUD/ AMERICA SAYS	2.0	4.8	2.0	May04/20	Rev #2: NZ
Contract Comment: FAMILY FEUD/ AMERICA SAYS																

Figure 6: Single page from document in DeepForm.

image quality). They were used to extend development and test set. The final dataset consists of 700 training, 100 development, and 300 test set documents. We rely on the standard F1 score for the purposes of DeepForm evaluation.

PWC★. The authors of AxCell relied on PWC Leaderboards and LinkedResults datasets [26]. The original formulation assumes extraction of $(task, dataset, metric, model, score)$ tuples from a provided table. In contrast, we reformulate the task as Document Understanding and provide a complete paper as input instead. These are obtained using arXiv identifiers available in the PWC metadata. Consequently, the resulting task is an end-to-end Key Information Extraction from real-world scientific documents.

Whereas LinkedResults was annotated consistently, the PWC is of questionable quality as it was obtained from leaderboards filled by Papers with Code visitors without a clear guideline or annotation rules. The difference between the two is substantial, i.e., the agreement in terms of F1 score between publications present in both PWC and LinkedResults is lower than 0.35. We attribute this mainly to flaws in the PWC dataset, such as missing records, inconsistent normalization and the difficulty of the task itself.

Consequently, we decided to perform its manual re-annotation assuming that: (1) The best result for a proposed model variant on the single dataset has to be annotated, e.g., if two models with different parameter sizes were present in the table, we report only the best one. (2) Single number is preferred (we take the average over multiple split or parts of the dataset if possible). (3) When results from the test set are available, we prefer them and don't report results from the validation set. (4) We add multiple value variants when possible. (5) We include information on used validation/dev/test split in the dataset description wherever applicable. (6) We don't report results on the train set. (7) We don't annotate results not appearing in the table. (8) We filter out publications that are hard to annotate even for a human.

Interestingly, human scores on PWC are relatively low in terms of ANLS value. This can be attributed to unrestricted nature of particular properties, e.g., *accuracy* and *average accuracy* are equally valid metric values. Similarly, *Action Recognition*, *Action Classification*, and *Action Recognition* are equally valid task names. We mitigated this problem by using ANLS-like comparison used in the F1 metric and providing multiple acceptable value variants, i.e., it is enough to provide half of the string representing one of the valid answers.⁷

Nevertheless, it is impossible to provide all answer variants during the preparation of the gold standard. We decided to keep the dataset in the benchmark as it is extremely demanding, and there is still a large gap between humans' and models' performance (See Table 3).

⁷Please refer to the metric implementation in the Github repository for a detailed description.

As the expected answer in PWC consists of a list of groups (property tuples that represent a complete record of the method, dataset, and results), the F1 metric here has to take into account the miss-placement of properties in another group. We assume the value is incorrect if placed in the wrong group (see reference implementation in supplementary materials).

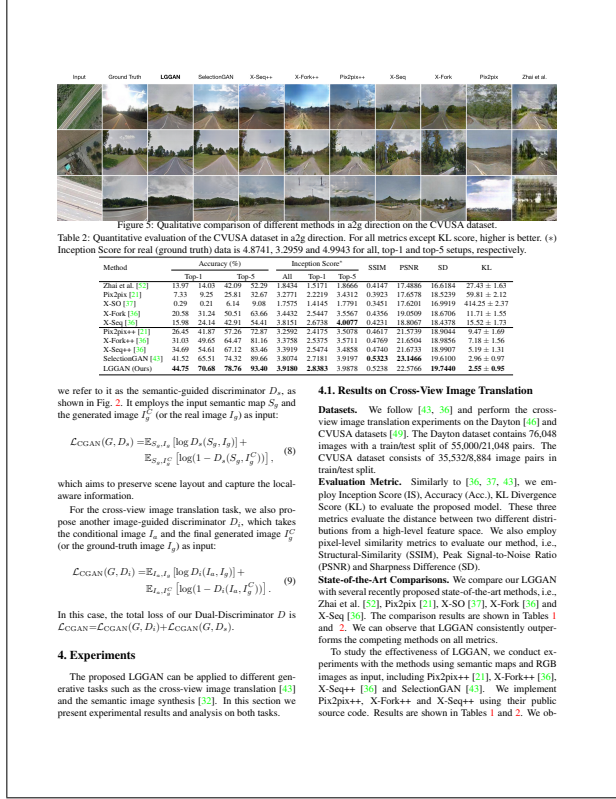


Figure 7: Single page from document in PWC.

D Dataset statistics

Chosen datasets represent the plethora of domains, lengths, and document types. This appendix covers the critical aspects of particular tasks at the population level.

Though part of the datasets is limited to one-pagers, the remaining documents range from a few to few hundred pages (Figure 8). At the same time, there is a great variety in how much text is present on a single page – we have both densely packed scientific documents and concise document excerpts or infographics. This diversity allows us to measure the ability to comprehend documents depending on their length.

E Details of human performance estimation

Estimation of human performance for PWC, WikiTableQuestions, DeepForm was performed in-house by professional annotators who are full-time employees of Applica.ai. Before approaching the process, each of them has to participate in the task-specific training described below.

Number of annotated samples depended on task difficulty and the variance of the resulting scores. We relied on 50 fully annotated papers for the PWC dataset (approx. 150 tuples with five values each), 109 DeepForm documents (532 values), and 300 questions asked to different WikiTableQuestion tables.

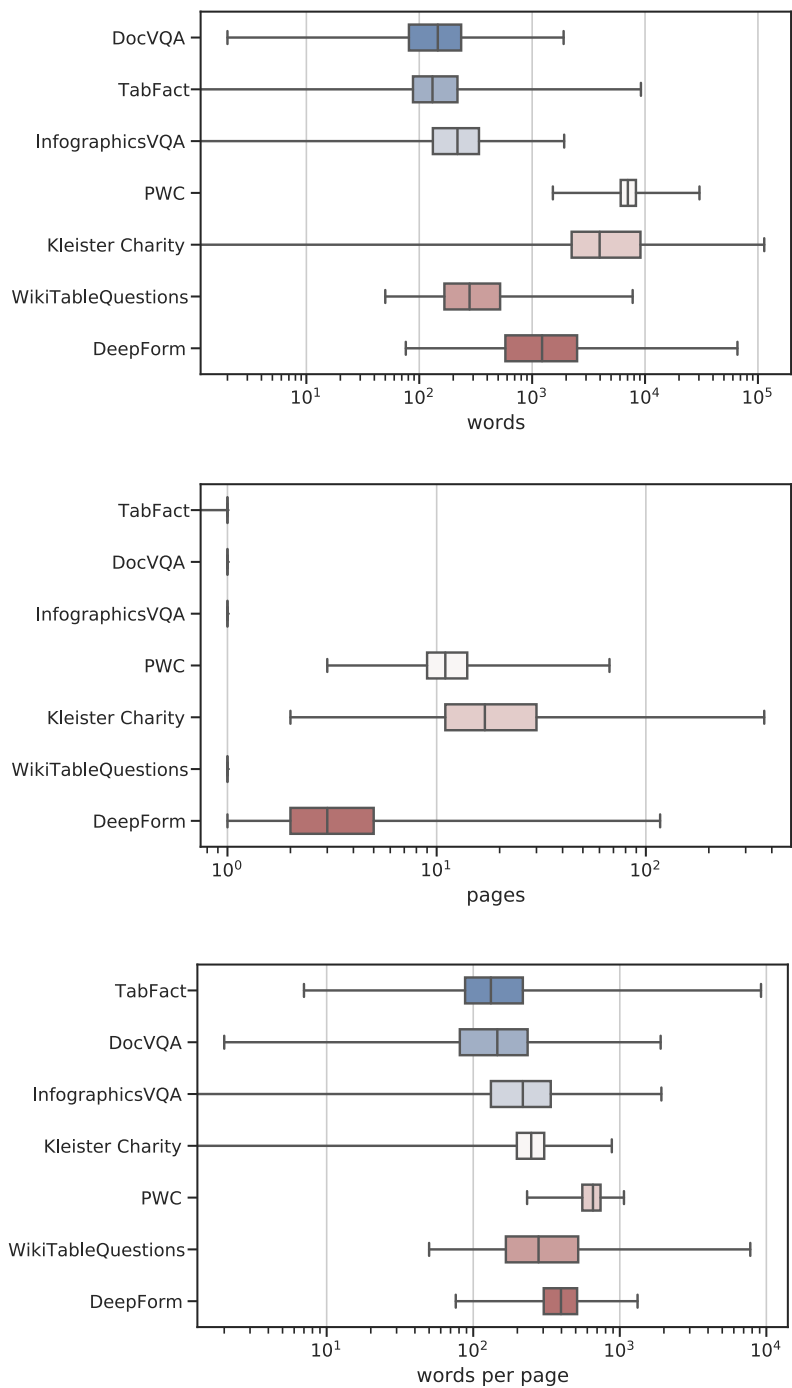


Figure 8: Number of words, pages, and words per page in particular datasets (log scale). Part of the datasets consist only of one-pagers.

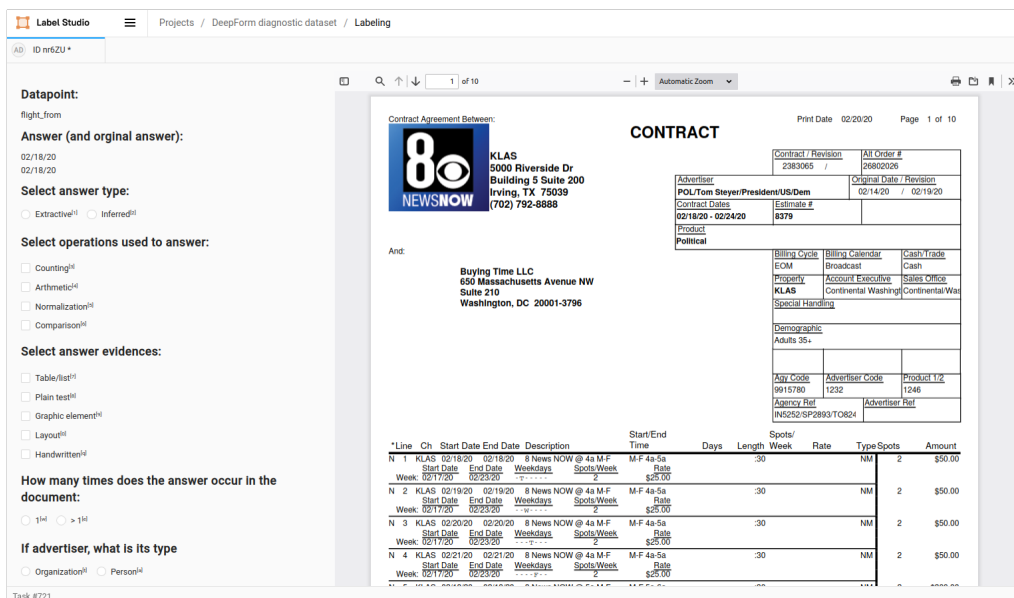


Figure 9: An example of an interface for annotating diagnostic subsets based on document from DeepForm dataset.

Each dataset was approached with two annotators in the LabelStudio tool. Human performance is the average of their scores when validated against the gold standard.

Training. Each person participating in the annotation process completed the training consisting of four stages: (1) Annotation of five random documents from the task-specific development set. (2) Comparative analysis of differences between their annotations and the gold standard. (3) Annotation of ten random documents from the task-specific development set and subsequent comparative analysis. (4) Discussion between annotators aimed at agreeing on the shared, coherent annotation rules.

F Annotation of diagnostic subsets

In order to analyze the prepared benchmark and the results of individual models, diagnostic sets were prepared. These diagnostic sets are subsets of examples selected from the testset for all datasets.

When building a taxonomy for diagnostic sets, we adopted two basic assumptions: (1) It must be consistent across all selected tasks so that at least two tasks can be noted with a given category (2) It should include as many aspects as possible that are relevant from the perspective of document understanding problem.

Initially, we adopted the taxonomies proposed in DocVQA, Infographics, and TabFact as potential categories [33, 32, 7]. In the next step, we adjusted our taxonomy to all datasets following the previously adopted assumptions, distinguishing seven main categories with 25 subcategories (for a more detailed description of the category (see the section F.1). Then, for each dataset, we prepared an annotation task in the LabelStudio tool ⁸ (see example 9) along with an annotation instruction. Finally, to determine Human performance, the annotation was carried out by a team of specialists from Applica.ai, where the selected example was noted only by one person.

F.1 Taxonomy description

The taxonomy is based on multiple aspects of documents, inputs, and answers and was designed to be sufficiently generic for future adaptation to other tasks. Here, in each category, we describe the predicates that annotators followed when classified an example into specific subcategories.

⁸<https://labelstud.io/>

Answer source. This category is based on the relation between answer and text in the document.

- Extractive – after lowercasing and white-characters removing, the answer can be exact-matched in the document.
- Inferred – other non-extractive cases.

Output format This category is based on the shape of an output.

- Single value – the answer consists of only one item.
- List – multiple outputs are to be provided.

Output type. This category is based on the semantic of an output.

- Organization – the answer is a name of an organization or institution.
- Location – the answer is a geographic location globally (e.g., a country, continent, city) or locally (building or street, among others).
- Person – the answer is a personal identifier(name, surname, pseudonym) or its composition. It can have a title prefix or suffix (e.g., Mrs., Mr., Ph.D.) or have a shortened or informal version.
- Number – numerical values given with the unit or percent. Values written in the free text do not comply with this class's definition.
- Date/Time/Duration – the answer represents the date, time, or the difference between two dates or times.
- Yes/No – the answer is a textual output of binary classification, such as Yes/No pairs, and Positive/Negative, 0/1 among others.

Evidence. This category is based on the source of information that allows the correct answer to be generated. When there are multiple justifications based on different pieces of evidence (for example, the address is in a table and block text), it is required to select all the pieces of evidence.

- Table or List – a *table* is a fragment of the document organized into columns and rows. The distinguishing feature of the table is consistency within rows and columns (usually the same data type). Moreover, it may have a header. In that sense, the form is not a table (or at least it does not have to be). A *list* is a table degenerated into one column or row containing a header.
- Plain text – the answer is based on plain text if there is an immediate need to understand a longer fragment of the text while answering.
- Graphic element – the answer is based on graphic evidence when understanding graphically rich, non-text fragments of documents (e.g., graphics, photos, logos (non-text)) are necessary for generating a correct answer.
- Layout – it is evidence when comprehending the placement of text on the page (e.g., titles, headers, footers, forms) is needed to generate the correct answer. This type does not include tables.
- Handwritten – when the text written by hand is crucial for an answer.

Operation. This category is based on the type of operations that are to be performed on the document before reaching to the correct answer.

- Counting – when there is a need to count the occurrences or determine the position on the list.
- Arithmetic – when there is an arithmetic operation applied before answering, or a sequence of arithmetic operations (e.g., averaging).
- Comparison – a comparison in the sense of lesser/greater. Other procedures that a comparison operation can express (e.g., approximation) may be chosen. Here, the operation "is equal" is not a comparison since it is sufficient to match sequences without a semantic understanding.
- Normalization – when we are to return something in the document but in a different form. It may only apply to the output; we do not acknowledge this operation when it is required to normalize a question fragment to match it in the document.

Answer number. This category is based on the number of occurrences of an answer in the document.

- 1 – when there is one path of logical reasoning to find the correct answer in the document. We treat it as one justification for two different reasoning paths based on the same data from the document.
- > 1 – the other cases.

G Unified format

We propose a unified format for storing information in the Document Understanding domain and deliver converted datasets as part of the released benchmark. It assumes three interconnected levels: dataset, document-annotation and document-content. Please refer to the repository for examples and formal specifications of the schemes.

Dataset. The dataset level is intended for storing the general metadata, e.g., name, version, license, and source. Here, the JSON-LD format based on the well-known schema.org web standard is used.⁹

Document. The documents annotation level is intended to store annotations available for individual documents within datasets and related metadata (e.g., external identifiers). Our format, valid for all the Document Understanding tasks, is specified using the JSON-Schema standard. This ensures that every record is well-documented and makes automatic validation possible. Additionally, to make the processing of large datasets efficient, we provide JSON Lines file for each split, thus it is possible to read one record at a time.

Content. As part of the original annotation or additional data we provide is related to document content (e.g., the output of a particular OCR engine), we introduce the document’s content level. Similarly to the document level, we propose an adequate JSON Schema and provide the JSON Lines files in addition. PDF files with the source document accompany dataset -, document-, and content-level annotations. If the source PDF was not available, a lossless conversion was performed.

H Evaluation protocol

Evaluation protocol. All the benchmark submissions are expected to conform to the following rules to guarantee fair comparison, reproducibility, and transparency:

- All results should be automatically obtainable starting from either raw PDF documents or the JSON files we provide. In particular, it is not permitted to rely on the potentially available source file that our PDFs were generated from or in-house manual annotation.
- Despite the fact that we provide an output of various OCR mechanisms wherever applicable, it is allowed to use software from outside the list. In such cases, participants are highly encouraged to donate OCR results to the community, and we declare to host them along with other variants. It is expected to provide detailed information on used software and its version.
- Any dataset can be used for unsupervised pretraining. The use of supervised pretraining is limited to datasets where there is no risk of information leakage, e.g., one cannot train models on datasets constructed from Wikipedia tables unless it is guaranteed that the same data does not appear in WikiTableQuestions and TabFact.
- It is encouraged to use datasets already publicly available or to release data used for pretraining.
- Training performed on a development set is not allowed. We assume participants select the model to submit using training loss or validation score. We do not release test sets and keep them secret by introducing a daily limit of evaluations performed on the benchmark’s website.
- Although we allow submissions limited to one category, e.g., QA or KIE, complete evaluations of models that are able to comprehend all the tasks with one architecture are highly encouraged.
- Since different random initialization or data order can result in considerably higher scores, we require the bulk submission of at least three results with different random seeds.

⁹See <https://json-ld.org/> for information on the JSON-LD standard, and <https://developers.google.com/search/docs/data-types/dataset> for the description of adapted schema.

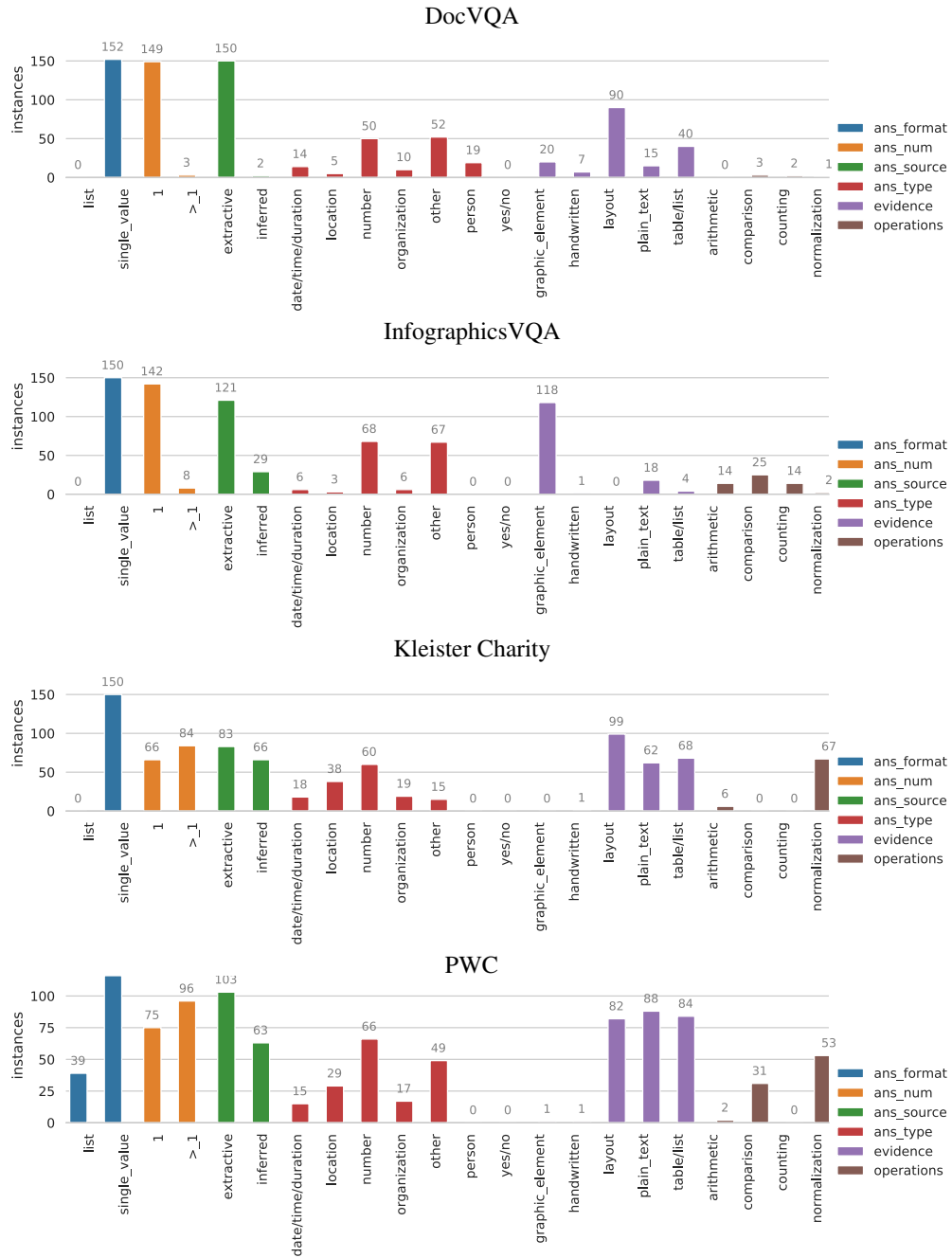


Figure 10: Number of annotated instances in each diagnostic subset category. DocVQA, InfographicsVQA, Kleister Charity, and PWC considered separately.

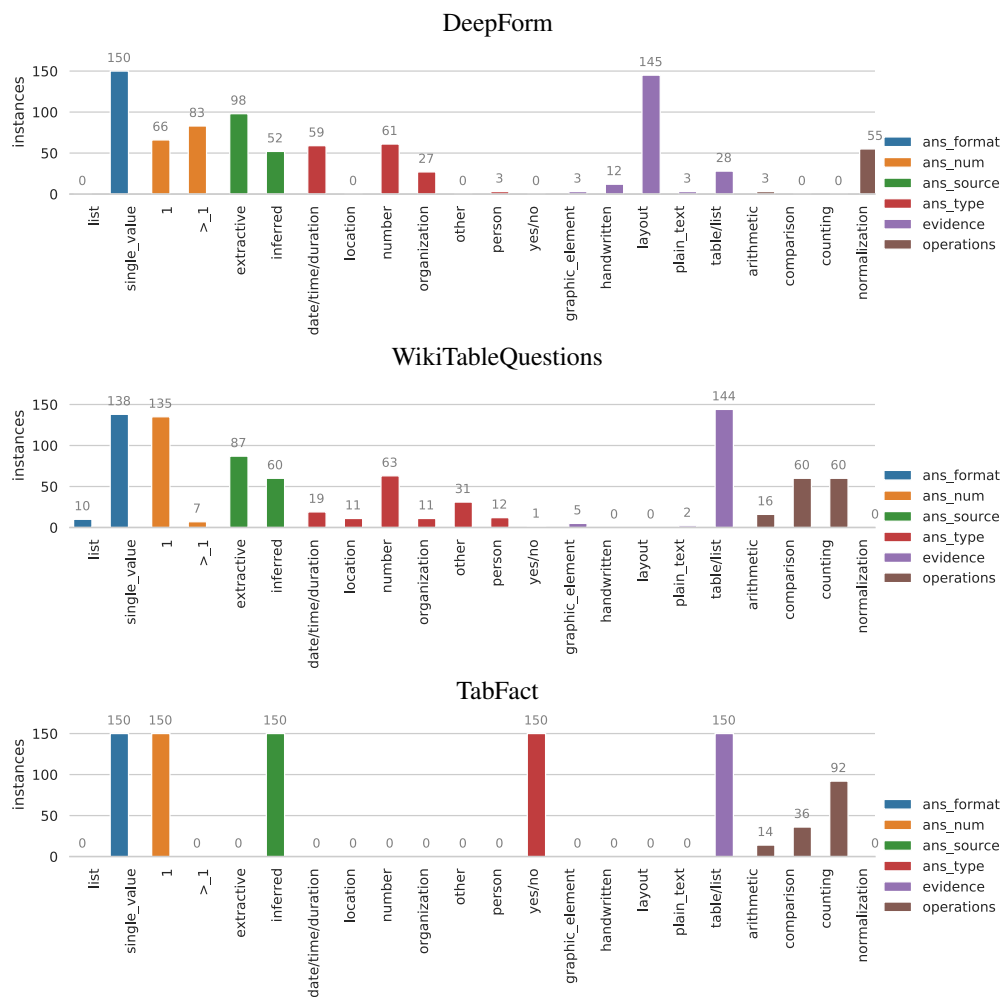


Figure 11: Number of annotated instances in each diagnostic subset category. DeepForm, WikiTableQuestions, and TabFact considered separately.

- Every submission is required to have an accompanying description. It is recommended to include the link to the source code.

I Experiments — training details

The experiments were carried out in an environment with NVIDIA A100-40G cards, PyTorch version 1.8.1, and the *transformers* library in version 4.2.2.

The parameters were selected through empirical experiments with T5-Base model on DocVQA and InfographicsVQA collections. The T5-Large model was used as the basis for finetuning.

The training lasted up to 30 epochs at batch 64 in training, the default optimizer AdamW ($\text{lr} = 2e-4$), and warmup set to 100 updates. Validation was performed five times per epoch, and when no improvement was seen for 20 validation steps (4 epochs), the training was stopped. The length of the input documents has been truncated to 6144 tokens for all datasets (only Kleister Charity and PWC benefited from that change, for the rest of them 1024 tokens is sufficient)¹⁰ and the responses to 256 tokens. Dropout was set to 0.15, gradient clipping to 1.0, and weight decay to $1e-5$.

¹⁰The hard limit of 6k tokens results from the memory limitation of the used GPU.



Arxiv Tables: Document Understanding Challenge Linking Texts and Tables

Karolina Konopka¹, Michał Turski^{1,2(✉)}, and Filip Graliński^{1,2}

¹ Adam Mickiewicz University, Poznań, Poland
karkon13@st.amu.edu.pl, {mturski, filipg}@amu.edu.pl

² Applica.ai, Warsaw, Poland
{michal.turski, filip.gralinski}@applica.ai

Abstract. We introduce Arxiv Tables, a novel challenge for Document Understanding focused on tables, but in relation to text passages. In order to build the data set, we leverage arXiv, a large open-access archive of scholarly papers. We use both \LaTeX source codes and graphical renderings of papers and combine tables with their references in the main text to create a quasi-Question Answering dataset by masking selected fragments available in the table. What distinguishes the dataset is that (1) the domain is science, (2) the input texts are longer than in typical Document Understanding Question Answering tasks, and (3) both the input and output contain non-standard characters used in scientific notation. For easier comparison for future research using this dataset, strong baselines are also given.

Keywords: Document understanding · OCR · Question Answering · table processing

1 Introduction

Recently, there has been a lot of progress in the relatively new domain of Document Understanding encompassing tasks such as classification, information extraction, question answering done for documents of rich layout and graphical structures, including elements such as tables, graphs, listings, formulae. A number of Transformer-based models were proposed for processing documents, such as LayoutLM [30], LAMBERT [9], TILT [23], also including end-to-end models, i.e. working without assuming an external OCR module, such as Donut [16], Dessurt [7], or Pix2Struct [17].

What is even more fundamental is that a number of Document Understanding datasets, challenges, and benchmarks have been proposed, examples include SROIE [13], Kleister [26], DUE [2]. Question Answering challenges are of special interest as they fit the generative nature of modern language models, see e.g. the DocVQA challenge [19]. It still seems that opportunities offered by some raw

K. Konopka—Work done as master thesis.

data sets have not been fully exploited. In this paper, we are using data available at arxiv.org, open-access archive for over 2.1M scholarly articles to create a novel Document Understanding task related to tables.

The advantage of arXiv is that not only final documents (mostly in the PDF format), but also their sources (mostly in \LaTeX) are available. \LaTeX is a rich language that can express not only text formatting but also tables and references between them and the main text (the `\label` and `\ref` commands). We leveraged this data to create a quasi-QA task related to tables, called Arxiv Tables.

The main contributions of this paper are as follows:

- we prepared and publicly shared a new large Document Understanding task¹,
- the evaluations on a number of non-trivial baselines were carried out².

2 Related Work

A number of challenges or even benchmarks (see e.g. the DUE benchmark [2]) has been proposed for Document Understanding. Some of them are in the Question Answering (QA) setup and one of the most popular is DocVQA [19], in which the documents were sourced from UCSF Industry Documents Library, “a digital archive of documents created by industries that influence public health”, in general, is a popular source of documents for Document Understanding challenges. The questions for DocVQA were crowd-sourced, the average length of a question is only 8.12 words and about 1/4 of the questions are of ‘table/list’ type.

2.1 Tabular Question Answering

There are also a few QA datasets with tabular input. BioTABQA [18] is a Question Answering dataset in a biomedical domain. The authors used templates to create questions and answers to tables from a medical textbook. As all the tables have the same format, a template approach was applicable. Unfortunately, the dataset diversity is small: there are only 22 question-and-answer templates and only one table format. WikiTableQuestions [22] is a dataset of crowd-sourced question-and-answer pairs to randomly selected 2,108 Wikipedia tables. Another work in the domain of QA using tables from Wikipedia is HybridQA [5]. The authors created the dataset, where a table is contextualized by text, namely they used text passages linked by hyperlinks from the table cells. Question and answer pairs to these table-and-text examples were obtained by crowd-sourcing.

Arxiv Table dataset differs from the one mentioned about by domain – there has not been any QA or quasi-QA dataset using tables for the scientific domain.

¹ The dataset is available at Gonito.net platform: <https://gonito.net/challenge/arxiv-tables>.

² Scripts and reproduction instructions are available at <https://github.com/applicaii/arxiv-tables-baselines>.

This particular domain is challenging because of specific terminology and non-standard characters present in scientific texts, especially in tables with measurements. Our dataset is also bigger, both in terms of tables and number of examples (see Table 1).

Table 1. Comparison of different Question Answering and quasi-Question Answering tasks in domain of table understanding.

Dataset	Domain	Tables #	Examples #
BioTABQA	Biomedical	513	31k
WikiTableQuestions	Wikipedia	2k	22k
HybridQA	Wikipedia	13k	70k
Arxiv Table (ours)	Scientific	96k	127k

2.2 Table to Text

There is another line of work in Document Understanding called Table to Text. In this paradigm, the goal is to generate a textual description of a table. One of the most popular datasets of this kind is ROTOWIRE [29], where a model is required to generate a summary of an NBA match using statistics about it presented in a form of a table. Another dataset is LogicNLG [4], where the inputs are tables from Wikipedia and the goal is to “generate natural language statements that can be logically entailed by the facts in the table”. The statements to be entailed were written by crowd-workers. NummericNLG [27] is a dataset from a scientific domain, where the source is articles from ACL Anthology website³. The tables are automatically extracted from the articles and the expected texts are obtained by a heuristic, which matches the table to its description in the text using the table’s reference number. SciXGen [3] applies a very similar idea to scientific articles from arXiv, but model input is contextualized by a text before the reference. SciGen [20] is also a dataset of arXiv tables, but the expected output texts are table descriptions, which were manually written by experts in a particular scientific domain. There are also two datasets, where the expected output is a summary only of a particular part of a table, and the remaining content of the table serves as a context. The first of them is WikiTableText [1], in which the goal is to generate a summary of a highlighted row from a table. The second one is ToTTo [21], in which a model is required to generate a summary of a few highlighted cells. Both datasets use Wikipedia as a table source and the expected outputs were manually written.

The dataset presented in this paper differs from the one described in this section because of its character. While Table to Texts challenges are focused on generating text, the goal in our dataset is to infer one value. Because of that different metrics are used: Table to Text challenges are evaluated using text

³ <https://aclanthology.org/>.

generation metrics like perplexity or BLEU, while Arxiv Tables uses accuracy. Text generation metric put more focus on generating coherent, grammatically correct text, while accuracy score (in the way we use it) promotes solution which is the best at extracting data from a table.

2.3 Table Extraction

Table Extraction is a problem where the goal is to extract either table structure, contents, or both from an image of a table. This task has been addressed in a number of datasets, see, for instance, TabLex [8] and PubTables-1M [25] as recent examples of large datasets of this type. Tables generated from \LaTeX source codes were also included in these datasets. AxCell [15] (also known as PWC) is a dataset closely related to the table extraction problem, where the goal is to extract a machine learning leaderboard from a scientific paper. A model is required to present a leaderboard in a standardized manner, hence this task requires a model to perform some reasoning and normalization.

By comparison, Arxiv Tables is *not* a table extraction challenge, it is concerned more with understanding tables in the context of a text fragment.

3 Dataset

The Arxiv Tables dataset is based on articles published on the arXiv.org⁴ website, which provides publicly available archives of scientific documents. It collects articles in mathematics, physics, astronomy, electrical engineering, computer science, quantitative biology, statistics, mathematical finance, and economics, and many of these include tables (examples are given in Fig. 1).

Each instance in the Arxiv Tables dataset consists of an image of a table, the quasi-question (referring to a given table) with masked information, and the expected answer (see an example in Fig. 2).

3.1 Preparation

The source data of the articles, which consist of compressed packages with \LaTeX files, were collected in bulk from Amazon S3. Then the documents were automatically searched for the existence of tables and paragraphs that include a \LaTeX reference (`\ref{}` command) to a table.

In the last stage, each paragraph with reference had key information masked, and the tables were saved as images. Training, test, and validation sets were created with a structure suitable for the Gonito.net⁵ challenge [10].

Most of the packages are *TAR* archives with *GNU zip* the format or single files compressed to *GNU zip* format. All of them include \LaTeX files, which were submitted to arXiv.org to produce PDF files after automatic processing by

⁴ <https://arxiv.org/>.

⁵ <https://gonito.net/>.

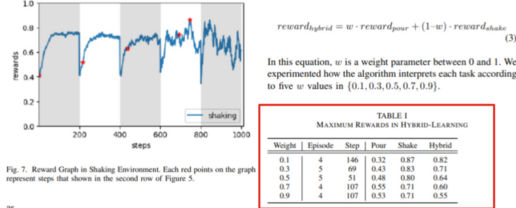


Fig. 7. Reward Graph in Shaking Environment. Each red point on the graph represent steps that shows in the second row of Figure 5.

$$r_{\text{shaking}} = \eta_{\text{pot}} / \eta_{\text{p}} \quad (2)$$

Through this, we trained the pot to conserve the maximum amount of water in the pot. We used the PPO algorithm like the pouring environment and trained the model for 5 episodes of 200 steps each.

1) *Quantitative Analysis:* Figure 7 shows the trend of reward as the learning progresses in the shaking environment. The reward tends to decrease slightly after reaching a saturation level in the first three episodes, which indicates that the agent has stuck in the local minimum during training. However, by solving the local minimum problem after episode 4 through exploration, the maximal reward increases. As a result, compared to the initial state which saves only 41% of the water, at the end of the training, the resultant pot is able to keep 86% of the water. The bottom right image of the shaking environment in Figure 5 is when the algorithm gets a reward of 0.86, as recorded in episode 4 step 147.

2) *Qualitative Analysis:* Looking at the changes in the models generated by the reinforcement learning, you can see how the network is trained to protect water. As you can see from the bottom row of Figure 5, the bottom part of the pot becomes larger and larger to keep as much water as possible, and the structure is good for storing water. As training proceeds, it was difficult to store water in the lower part, and the training progressed with a double tube structure. Commonly, there is a barrier structure in the upper part to prevent the pot from splashing by water shaking. Consequentially, we were able to confirm that when the simulation was carried out, it was trained to keep the bouncing water as much as possible in the pot from the large swing of ± 70 degrees.

D. Hybrid-Learning

In hybrid-learning, we examined the possibility of pot design that can perform both contradictory tasks. To do this, we defined a new reward which is a weighted sum of the reward in the pouring environment and the reward in the shaking environment as follows:

$$r_{\text{shaking}} = w \cdot r_{\text{reward}_{\text{pour}}} + (1-w) \cdot r_{\text{reward}_{\text{shake}}} \quad (3)$$

In this equation, w is a weight parameter between 0 and 1. We experimented how the algorithm interprets each task according to five w values in $\{0.1, 0.3, 0.5, 0.7, 0.9\}$.

TABLE I
MAXIMUM REWARDS IN HYBRID-LEARNING

Weight	Episode	Step	Pour	Shake	Hybrid
0.1	4	146	0.32	0.87	0.82
0.3	5	69	0.43	0.83	0.71
0.5	5	51	0.48	0.80	0.64
0.7	4	107	0.55	0.71	0.66
0.9	4	107	0.53	0.71	0.55

1) *Quantitative Analysis:* Table I indicates how much reward is obtained for different weight parameters. Each episode and step indicates the time when the maximum hybrid reward was achieved for the corresponding weight, and the three rewards (pouring, shaking, and hybrid) are the corresponding rewards at the time. When w is 0.1, according to (3), we can see that the shaking environment has more weight on training. As w increases, training is more focused on the pouring environment. As can be seen in Table I, in the pouring environment, a pouring reward of 0.32 was achieved at the point where the hybrid reward was largest when w was 0.1. As w increased to 0.9, the pouring reward increased to 0.53, which is the best score of the single pouring environment, because the algorithm gave more weight to the pouring environment. Conversely, in the shaking environment, the shaking reward was 0.87 when w was 0.1 and the reward decreased to 0.71 when w was 0.9. In this way, we showed the deep RL algorithm combining the two opposite tasks can train a model that satisfies both tasks.

2) *Qualitative Analysis:* Figure 8 shows how the model appears based on the change in the weight parameter w . When the value of w is 0.1, there is a water trap structure at lower position, a narrow entrance, and a barrier structure below the entrance like the model designed in the pure shaking environment. This shows that the training is focused on the shaking environment and trained to maximize water in the pot. On the other hand, when w is 0.9, we can see that the model has trained to create a smooth line in the middle like the model designed in the pouring environment and flows the water as easily as possible. In the case of the third model of $w = 0.5$, which performs the two tasks in the most balanced way, we can see that the model design maintains all of these features. Though there exists a storage part in the middle influenced by the shaking environment, it has a tendency to minimize the water remaining in the pot through the narrowing structure from the bottom to the top which resembles the design of the pouring environment.

TABLE II ALGORITHM TO SOLVE (P1) WITH FINITE γ	
1.	Initialize $\mathbf{P} = 0$, $(\tau^{(0)}, \mathbf{P}^{(0)}) = (\tau^*, \mathbf{P}^*)$ obtained by solving (P1).
2.	Repeat
1)	Set $k = k + 1$.
2)	Set λ, μ and $\mathbf{P}^{(k-1)}$, solve (P4) by Proposition 3.2.
3)	Update λ and μ using the ellipsoid method and sub-gradient of $G^{(\mathbf{P}^{(k-1)}, \lambda, \mu)}$ given by (42) and (43), and obtain τ^* .
4)	Set $\tau^{(k)} = \tau^*$.
5)	Obtain $\mathbf{P}^{(k)}$ using $\mathbf{P}^{(k-1)}$, $\tau^{(k)}$, (36), and (44)-(47).
3.	Until $\ \tau^{(k)} - \tau^{(k-1)}\ , \ \mathbf{P}^{(k)} - \mathbf{P}^{(k-1)}\ \leq \epsilon$, (τ^*, \mathbf{P}^*) .

ellipsoid method [34] with the sub-gradient of $G^{(\mathbf{P}^{(k-1)}, \lambda, \mu)}$, denoted as $\tau = [\tau_1, \dots, \tau_K]^T$, given by

$$v_{\lambda} = \sum_{i=0}^K \tau_i - 1, \quad (42)$$

$$v_{\mu} = \sum_{i=0}^K P_i^{(k-1)} \tau_i - P_{\text{avg}}, \quad (43)$$

where τ_i , $i = 0, \dots, K$, are given by (40). The optimal solution of (P4), denoted by τ^* , is then obtained corresponding to the optimal dual solution of λ^* and μ^* after the ellipsoid method converges, and finally we have $\tau^{(k)} = \tau^*$ for given $\mathbf{P}^{(k-1)}$.

Once $\tau^{(k)}$ is obtained, $\mathbf{P}^{(k)}$ at the k -th iteration can be obtained using the obtained $\tau^{(k)}$. Denote $\mathbf{P}^{(k)}$ at the k -th iteration as $\mathbf{P}^{(k)} = [P_1^{(k)}, \dots, P_K^{(k)}]^T$, where $\mathbf{P}^{(k)} = [P_1^{(k)}, \dots, P_K^{(k)}]^T$. Since $\tau_i^{(k)} = 0$, $i \geq 1$, as shown in (40), we can thus take any value for $P_i^{(k)}$ such that $0 \leq P_i^{(k)} \leq P_{\text{max}}$. Furthermore, $\mathbf{P}^{(k)}$ can be obtained by applying the gradient projection method based on (36), as follows:

$$\tilde{\mathbf{P}}^{(k)} = \mathbf{P}_2 \left(\mathbf{P}^{(k-1)} + \delta^{(k)} \nabla_{\mathbf{P}} W_{\text{avg}}(\tau^{(k)}, \mathbf{P}^{(k-1)}) \right), \quad (44)$$

$$\mathbf{P}^{(k)} = \mathbf{P}^{(k-1)} + \delta^{(k)} \left(\tilde{\mathbf{P}}^{(k)} - \mathbf{P}^{(k-1)} \right), \quad (45)$$

where $\delta^{(k)} \in (0, 1]$ and $\delta^{(k)}$ are both small step sizes. In addition, $\nabla_{\mathbf{P}} W_{\text{avg}}(\tau^{(k)}, \mathbf{P}^{(k-1)}) = [g_1^{(k)}, \dots, g_K^{(k)}]^T$ denotes the gradient of $W_{\text{avg}}(\tau^{(k)}, \mathbf{P}^{(k-1)}) \triangleq \sum_{i=1}^K R_i^{(k)}(\tau_i^{(k)}, P_i^{(k-1)})$ with $g_i^{(k)}$, $i = 1, \dots, K$, given by

$$g_i^{(k)} = \frac{\partial R_i^{(k)}(\tau_i^{(k)}, P_i^{(k-1)})}{\partial P_i^{(k-1)}} + \gamma \left(P_{\text{avg}} - \sum_{i=1}^K P_i^{(k-1)} \right) \quad (46)$$

Furthermore, \mathcal{E} denotes the feasible set of \mathbf{P} given $\tau^{(k)}$ and $P_i^{(k)}$, defined by

$$\mathcal{E} = \left\{ \mathbf{P} \mid \sum_{i=1}^K P_i = P_{\text{avg}}, 0 \leq P_i \leq P_{\text{max}}, i = 1, \dots, K \right\} \quad (47)$$

Finally, in (44) $\mathbf{P}_2(\mathbf{x})$ denotes the operation of projection of \mathbf{x} onto \mathcal{E} . To summarize, one algorithm to solve problem (P1) with finite γ is given in Table II.

At each iteration of the algorithm given in Table II, the computational complexity of step 2.2) is $\mathcal{O}(K)$. Similarly to the case with perfect SIC in the previous subsection, $\mathcal{O}(K)$ computations are required for step 2.3). In step 2.5), $\mathcal{O}(K)$, $\mathcal{O}(K)$, and $\mathcal{O}(K)$ computations are required for computing $\nabla_{\mathbf{P}} W_{\text{avg}}(\tau^{(k)}, \mathbf{P}^{(k-1)})$ in (46), $\mathbf{P}^{(k)}$ in (44), and $\mathbf{P}^{(k)}$ in (45), respectively. Therefore, the total time complexity of the algorithm in Table II is $\mathcal{O}(K^2)$.

Remark 3.1: With $P_{\text{max}} \rightarrow \infty$, we can choose $P_i^* \rightarrow \infty$ such that $\tau_i^* P_i^* \rightarrow P_{\text{avg}}$ resulting in $P_i = 0$, $i = 1, \dots, K$. In this special case, the FD-WPCN with finite SI is in fact equivalent to that in the previous ideal case of perfect SIC with $P_{\text{max}} \rightarrow \infty$.

IV. OPTIMAL TIME AND POWER ALLOCATION IN HD-WPCN

In this section, we study the optimal time and power allocation in HD-WPCN to maximize the WSR. Given $P = \min\{P_{\text{avg}}/\tau_i, P_{\text{max}}\}$, from (16) the WSR maximization problem for HD-WPCN is formulated as

$$(P5): \max_{\tau} \sum_{i=1}^K \alpha_i R_i^{(k)}(\tau, P) \quad (48)$$

$$\text{s.t.} \quad \sum_{i=1}^K \tau_i \leq 1, \quad (49)$$

$$\tau_i \leq P_{\text{avg}}, \quad (50)$$

$$P \leq P_{\text{max}}, \quad (51)$$

$$P \geq 0, \tau_i \geq 0, i = 0, 1, \dots, K.$$

Problem (P4) is non-convex in general due to the non-concave objective function defined in (16) and non-convex average power constraint in (49). To solve problem (P4), we first consider the following WSR maximization problem for HD-WPCN when the transmit power of the H-AP is fixed as $P = P_{\text{max}}$:

$$(P6): \max_{\tau} \sum_{i=1}^K \alpha_i \tau_i \log_2 \left(1 + \alpha_i P_{\text{max}} \frac{\tau_i}{\tau_i} \right) \quad (52)$$

$$\text{s.t.} \quad \sum_{i=0}^K \tau_i \leq 1, \quad (53)$$

$$\tau_i \geq 0, i = 0, 1, \dots, K,$$

with α_i defined in (18). As shown in (21), (P5) is convex and its optimal time allocation solution, denoted by $\tau^* = [\tau_1^*, \dots, \tau_K^*]^T$, is given in the following lemma.

Lemma 4.1: For problem (P6), the optimal solution is given by

$$\tau_i^* = \frac{1}{1 + P_{\text{max}} \sum_{j=1}^K (\alpha_j / \alpha_i)}, \quad (54)$$

$$\tau_i^* = \frac{P_{\text{max}} \alpha_i / \alpha_i}{1 + P_{\text{max}} \sum_{j=1}^K (\alpha_j / \alpha_i)}, \quad i = 1, \dots, K, \quad (52)$$

Fig. 1. Two different examples of pages from documents published on arXiv.org with tables marked. The table on the left contains numeric data [6] and the table on the right has one row that contains a description of an algorithm [14].

AutoTeX software. It should be noted that the Amazon S3 storage sometimes includes PDF files instead of source files and we filtered them out since they could not be automatically processed.

Each compressed file or archive was automatically checked for the presence of tables and \LaTeX references. It is worth noting that only those paragraphs that referred to exactly one table were used for the final set to avoid ambiguity (the same value or name may appear in multiple tables, it's hard to say which table is the right context for a particular value). However, one table may have multiple \LaTeX references that are used in the dataset. After extraction of every paragraph that includes \LaTeX reference to only one table and creating images of the tables, the paragraph was automatically converted into a quasi-question that has one key information masked and the masked key information becomes an expected answer.

Expected answer in our dataset is a specific piece of information from a table, either:

- a text containing between 3 and 15 characters present explicitly in the table (an ‘extractive’ example),
- or one of the eight listed adjectives: **largest**, **smallest**, **better**, **worse**, **best**, **worst**, **more**, **less** (‘Comparative’ examples⁶; they do *not* have to be present in the table, the assumption is that we can check deeper reasoning capabilities of a model, not just ability to carry out information extraction, this way).

⁶ On the evaluation platform they are denoted as ‘degree’.

Weight	Episode	Step	Pour	Shake	Hybrid
0.1	4	146	0.32	0.87	0.82
0.3	5	69	0.43	0.83	0.71
0.5	5	51	0.48	0.80	0.64
0.7	4	107	0.55	0.71	0.60
0.9	4	107	0.53	0.71	0.55

As can be seen in Table I, in the pouring environment, a pouring reward of `<mask>` was achieved at the point where the hybrid reward was largest when w was 0.1. As w increased to 0.9, the pouring reward increased to 0.53, which is the best score of the single pouring environment, because the algorithm gave more weight to the pouring environment.

1) *Quantitative Analysis:* Table I indicates how much reward is obtained for different weight parameters. Each episode and step indicates the time when the maximum hybrid reward was achieved for the corresponding weight, and the three rewards (pouring, shaking, and hybrid) are the corresponding rewards at the time. When w is 0.1, according to (3), we can see that the shaking environment has more weight on training. As w increases, training is more focused on the pouring environment. As can be seen in Table I, in the pouring environment, a pouring reward of 0.32 was achieved at the point where the hybrid reward was largest when w was 0.1. As w increased to 0.9, the pouring reward increased to 0.53, which is the best score of the single pouring environment, because the algorithm gave more weight to the pouring environment. Conversely, in the shaking environment, the shaking reward was 0.87 when w was 0.1 and the reward decreased to 0.71 when w was 0.9. In this way, we showed the deep RL algorithm combining the two opposite tasks can train a model that satisfies both tasks.

Fig. 2. An example of an image of a table [6] with text referring to it, along with a selected piece of information that can be masked because it is present in the table. In this case value 0.32 can be masked because it is referenced in the table.

However words such as `and`, `or`, `from`, `of`, `the`, `model`, `for`, `table`, were explicitly excluded as answers because of a lack of specificity.

3.2 Dataset Statistics

The total number of downloaded and correctly processed documents is 52883.

The resulting corpus, divided into a test, training, and validation sets, has the sizes presented in Table [2]. Expected answers in each subset contain around 25% of adjectives and 11% of numbers from the table, the remaining 64% are other kinds of entities. See Table 5 for some examples of expected answers.

The training subset has 3698 unique expected answers and the most frequent phrases are `more`(478), `better`(433), `best` (341), `less` (131), `largest` (82), `worse` (37), `smallest` (26), `worst` (25), 100 (22), 200 (17), AUC (12), `mean` (12), 0.5 (10), 300 (10), 1000 (10), LSTM (10), `time` (9), RMSE (9), `models` (9), CNN (9). 3035 values occur only once.

Table 3 presents statistics related to the length of quasi-questions. From these data, it can be concluded that most quasi-questions contain up to 500 words, on average they include almost 7 sentences, and quasi-questions that have around 13,000 words are an exception that is only present in the training set.

Table 2. Training, validation and test set sizes.

Subset	Quasi-questions #	Images #
train	114157	86978
test	6403	4932
validation	6039	4612

Table 3. Basic quasi-question and expected answer statistics for training, test and validation sets.

	Subset	Quasi-question			Expected answer
		characters	words	sentences	characters
MIN	train	8	4	1	3
	test	36	12	1	3
	validation	32	7	1	3
MAX	train	256382	12950	122	15
	test	18337	1250	50	15
	validation	11233	2234	84	15
MEAN	train	874.88	177.34	6.94	7.39
	test	864.74	175.16	6.85	7.06
	validation	868.69	176.32	6.94	7.11
MEDIAN	train	763	154	6	7
	test	763	154	6	6
	validation	761	154	6	6

3.3 Evaluation Procedure

The solutions are evaluated using GEval evaluation tool [11] using Accuracy evaluation metric. Values of Accuracy for the two subsets of items (adjectives and strings present in the tables) are also calculated as supplementary metrics.

4 Baseline Models

4.1 LayoutLMv3

The first of our baselines is LayoutLMv3 [12]. It is a Transformer-based [28] model for document understanding (mostly information extraction) tasks. As input to the model, we need an image of a document (in our case it is a table image), document tokens with bounding boxes, and a prompt. To produce tokens and bounding boxes of a table we used Microsoft Read v.3.2. The maximum length of a model input equals 512 tokens (for a quasi-question and document tokens together). Because of that, we decided to truncate a quasi-question to 25 words only, we use 17 words of context before [Mask], the [Mask] token, and 7

words of context after the [Mask]. This set-up was motivated by the intuition that the context before the [Mask] is more important, as it is the natural order of reading. We feed the model with all the tokens from the truncated quasi-question and as many tokens from the table, as it fits in a vector of size 512.

LayoutLMv3 uses a sequence-labeling approach, hence to calculate loss we need annotations at the token level. Because of that the proposed baseline can only work with ‘extractive’ examples. To provide labels of ‘extractive’ datapoints at a token level, we proposed a heuristic for token labeling. It finds the expected answer in the table using fuzzy matching. The whole pipeline of training pipeline is presented in Fig. 3.

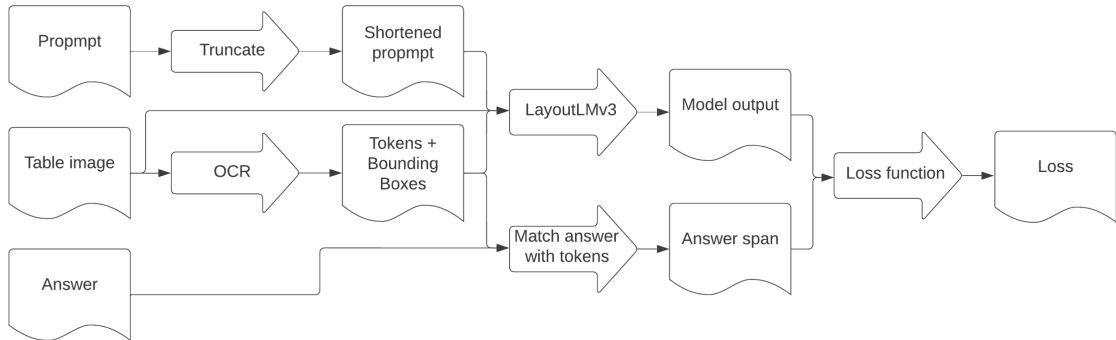


Fig. 3. Pipeline for training LayoutLMv3 baseline.

4.2 T5

We also provide some baselines in the sequence-to-sequence paradigm, namely T5 [24] and its adaptation for document understanding T5+2D [2]. Input to both models is a quasi-question together with tokens from a document. T5+2D uses also information about the bounding boxes of the tokens. Once again to provide tokens and bounding boxes we used Microsoft OCR v.3.2. The T5 family of models has no predefined maximum length of the input, hence the inputs we provided to the models were only limited by GPU memory (40GB), namely not more than 1450 tokens. Because of that, we did not truncate the quasi-question, an input was built out of the whole quasi-question and as many tokens from the table as possible. In more than 90% of cases, 1450 tokens input vector was long enough to contain the whole quasi-question and all the tokens from the table. As the model produces sequence as an output, we do not need any knowledge about exact answer span to calculate loss, we can just compare generated tokens with the expected answer. The whole pipeline is presented in Fig. 4.

Following the authors of the document understanding benchmark DUE [2], we trained 4 models using such a pipeline: the T5 model, the T5+2D model, which uses positions of tokens on a page, and their pretrained versions. T5 and T5+2D models are publicly available and pretrained ones were shared with us thanks to the courtesy of authors of the DUE benchmark.

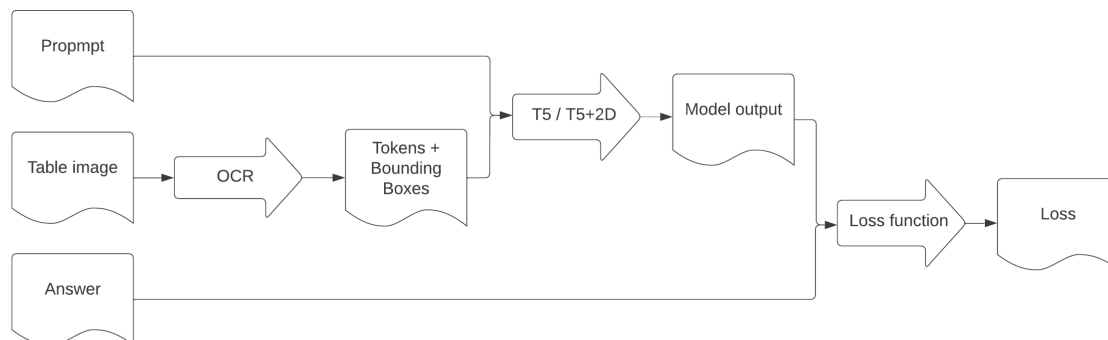


Fig. 4. Pipeline for T5 and T5+2D baselines. Bounding boxes are used only by the T5+2D model, plain T5 ignores them.

5 Results

Table 4. The detailed results (average accuracy over 3 runs) of our baselines on the test set. U stands for unsupervised pretraining. Value after \pm is standard deviation.

Datapoints	LayoutLMv3	T5	T5+2D	T5+U	T5+2D+U
Extractive	49.5 \pm 1.1	60.7 \pm 0.4	59.1 \pm 1.1	62.6 \pm 0.8	63.5 \pm 0.3
Comparative	—	86.4 \pm 0.5	86.0 \pm 0.3	86.0 \pm 0.4	85.1 \pm 0.2
All	—	66.9 \pm 0.4	65.6 \pm 0.9	68.3 \pm 0.7	68.7 \pm 0.3

The detailed results are presented in Table 4. The weakest of the presented models is LayoutLMv3. Its performance was probably limited by three factors: length of the input sequence, OCR mistakes, and quality of token labeling. Firstly, we were not able to feed the whole quasi-question and all the tokens from a table to the model, for some tables expected answers were not even present in the tokens provided. Secondly, the OCR engine used works well with Latin characters, but when the table contains other symbols, such as Greek letters or special characters denoting units of measurement, it fails. The sequence labeling model does not have any mechanism for correcting OCR mistakes: it may only compose an answer from the input tokens. Finally, the heuristic for token labeling introduces its own mistakes, especially when applied to noisy OCR output. The sequence labeling approach cannot be applied to datapoints other than ‘extractive’ one, hence ‘comparative’ and ‘all’ scores were not presented.

Models from the T5 family were also affected by OCR mistakes, but theoretically, sequence-to-sequence models can generate answers not present in the input and, thanks to that, correct OCR (even if the OCR engine returned wrong predictions). Unfortunately, the T5 dictionary is also limited in terms of special characters, hence sometimes the models have no capability of generating a correct answer. They cannot even read these kinds of characters and they internally

Table 5. Sample of 40 errors for random tables from the validation set for the T5+2D+U model.

Expected answer	Actual answer
SFR	log [LIR,SF]
R[O III]	D600
W80,max	W80
D600	max
(3)	(1)
(5)	v2
(6)	(1)
64×4	64 4
128×8	128 8
r0/a	ZP
largest	smallest
0.07	0.02
100	NS2
less	more
changes	users
ICD-11	iCAT
ϵ/σ	/0
$7 M_{\odot}$	= 0.5
best	bSNR
better	more
MNRM	coarser mesh
ML3	PGM3
$\text{const} - \sqrt{\epsilon}$	1/
best	smallest
s)	k + n2k
6.3	236.5
i = E*	i =
i=1 τ *	i=1
on-the-loop	mix
largest	smallest
slope	mass
SM(f1, f2, f3)	f' 3
OtW Φ	OtB
1998	1986
Ca i	Fel
Fe i	Fel
1988	X6708
Nijm I	AV18
more	less
0 1	1

represent them as special tokens [unknown]. As one may expect, pretrained models perform significantly better than their not pretrained counterparts. The not pretrained T5+2D model performed worse than the base T5 model. We hypothesize that it is caused by randomly initialized weights in layout related part of the model: they probably introduce more noise than useful information for the rest of the model. As far as the comparison between pre-trained T5 and T5+2D is concerned, there is not any significant difference in their performance. We think both models reached their glass ceiling caused by the level of OCR mistakes and limited dictionary, hence layout information does not help the T5+2D+U model to overcome these obstacles.

Note that the information about item types ('extractive' vs 'comparative') was not used during model training and testing, in particular models were not explicitly constrained to the eight adjectives/adverbs.

6 Error Analysis

A random sample of 40 errors for the current best (T5+2D+U) model is presented in Table 5. Six errors (15%) were directly caused by an OCR error (e.g. 64 4 instead of 64×4), four errors (10%) are the wrong forms of an adjective/adverb ('comparative' type, e.g. **smallest** instead of **largest**), in one case of the 'comparative' type a non-adjective/adverb were returned (**bSNR** instead of **best**).

7 Conclusions

By leveraging the availability of a large number of scientific papers and their source codes, we introduced a new, relatively large challenge for the domain of document understanding. The results of strong baselines still show that reasoning about a text in the context of tables requires new ideas for document understanding models, as even layout-aware models brought no improvement over their non-layout-aware counterparts.

In the future, a similar approach can be applied to figures/graphs (**figure** environment in \LaTeX) leading to a the quasi-Question-Answering challenge with even more pronounced visual aspects.

Acknowledgments. The Smart Growth Operational Programme partially supported this research under projects no. POIR.01.01.01-00-0144/17-00 (*Robotic processes automation based on Artificial Intelligence and deep neural networks*) and POIR.01.01.01-00-1624/20 (*Hiper-OCR - an innovative solution for information extraction from scanned documents*).

References

1. Bao, J., et al.: Table-to-text: describing table region with natural language. In: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 32, no. 1 (2018). <https://doi.org/10.1609/aaai.v32i1.11944>. <https://ojs.aaai.org/index.php/AAAI/article/view/11944>

2. Borchmann, L., et al.: DUE: End-to-end document understanding benchmark. In: Thirty-Fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2) (2021)
3. Chen, H., Takamura, H., Nakayama, H.: SciXGen: a scientific paper dataset for context-aware text generation. In: Findings of the Association for Computational Linguistics: EMNLP 2021, pp. 1483–1492. Association for Computational Linguistics, Punta Cana, Dominican Republic (2021). <https://doi.org/10.18653/v1/2021.findings-emnlp.128>. <https://aclanthology.org/2021.findings-emnlp.128>
4. Chen, W., Chen, J., Su, Y., Chen, Z., Wang, W.Y.: Logical natural language generation from open-domain tables. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 7929–7942. Association for Computational Linguistics, Online (2020). <https://doi.org/10.18653/v1/2020.acl-main.708>. <https://aclanthology.org/2020.acl-main.708>
5. Chen, W., Zha, H., Chen, Z., Xiong, W., Wang, H., Wang, W.: HybridQA: a dataset of multi-hop question answering over tabular and textual data. In: Findings of EMNLP 2020 (2020)
6. Choi, J., Hyun, M., Kwak, N.: Task-oriented design through deep reinforcement learning (2019). <https://doi.org/10.48550/ARXIV.1903.05271>. <https://arxiv.org/abs/1903.05271>
7. Davis, B., Morse, B., Price, B., Tensmeyer, C., Wigington, C., Morariu, V.: End-to-end document recognition and understanding with Dessurt. arXiv e-prints, pp. arXiv-2203 (2022)
8. Desai, H., Kayal, P., Singh, M.: TABLEX: a benchmark dataset for structure and content information extraction from scientific tables. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12822, pp. 554–569. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86331-9_36
9. Garncares, L., et al.: LAMBERT: layout-aware language modeling for information extraction. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12821, pp. 532–547. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86549-8_34
10. Gralinski, F., Jaworski, R., Borchmann, L., Wierzchon, P.: Gonito. net-open platform for research competition, cooperation and reproducibility. In: Proceedings of the 4REAL Workshop: Workshop on Research Results Reproducibility and Resources Citation in Science and Technology of Language, pp. 13–20 (2016)
11. Graliński, F., Wróblewska, A., Stanisławek, T., Grabowski, K., Górecki, T.: GEval: tool for debugging NLP datasets and models. In: Proceedings of the 2019 ACL Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP, pp. 254–262. Association for Computational Linguistics, Florence, Italy (2019). <https://www.aclweb.org/anthology/W19-4826>
12. Huang, Y., Lv, T., Cui, L., Lu, Y., Wei, F.: LayoutLMV3: pre-training for document AI with unified text and image masking. In: Proceedings of the 30th ACM International Conference on Multimedia (2022)
13. ICDAR: competition on scanned receipts OCR and information extraction (2019). <https://rrc.cvc.uab.es/?ch=13>. Accessed 01 Feb 2023
14. Ju, H., Zhang, R.: Optimal resource allocation in full-duplex wireless-powered communication network (2014). <https://doi.org/10.48550/ARXIV.1403.2580>. <https://arxiv.org/abs/1403.2580>
15. Kardas, M., et al.: Axccl: Automatic extraction of results from machine learning papers. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 8580–8594 (2020)

16. Kim, G., et al.: OCR-free document understanding transformer. In: Avidan, S., Brostow, G., Cissé, M., Farinella, G.M., Hassner, T. (eds.) ECCV 2022. LNCS, vol. 13688, pp. 498–517. Springer, Cham (2022). https://doi.org/10.1007/978-3-031-19815-1_29
17. Lee, K., et al.: Pix2Struct: screenshot parsing as pretraining for visual language understanding. arXiv preprint [arXiv:2210.03347](https://arxiv.org/abs/2210.03347) (2022)
18. Luo, M., Saxena, S., Mishra, S., Parmar, M., Baral, C.: Biotabqa: Instruction learning for biomedical table question answering. In: CEUR Workshop Proceedings, vol. 3180, pp. 291–304 (2022). Publisher Copyright: 2022 Copyright for this paper by its authors.; 2022 Conference and Labs of the Evaluation Forum, CLEF 2022; Conference date: 05–09-2022 Through 08–09-2022
19. Mathew, M., Karatzas, D., Jawahar, C.: DocVQA: a dataset for VQA on document images. In: Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV), pp. 2200–2209 (2021)
20. Moosavi, N.S., Ruckl'e, A., Roth, D., Gurevych, I.: Learning to reason for text generation from scientific tables. ArXiv abs/2104.08296 (2021)
21. Parikh, A., et al.: ToTTo: a controlled table-to-text generation dataset. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), pp. 1173–1186. Association for Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.emnlp-main.89>. <https://aclanthology.org/2020.emnlp-main.89>
22. Pasupat, P., Liang, P.: Compositional semantic parsing on semi-structured tables. In: Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing, vol. 1 (Long Papers), pp. 1470–1480. Association for Computational Linguistics, Beijing, China (2015). <https://doi.org/10.3115/v1/P15-1142>. <https://aclanthology.org/P15-1142>
23. Powalski, R., Borchmann, L., Jurkiewicz, D., Dwojak, T., Pietruszka, M., Pałka, G.: Going full-TILT boogie on document understanding with text-image-layout transformer. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12822, pp. 732–747. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86331-9_47
24. Raffel, C., ET AL.: Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **21**(140), 1–67 (2020). <http://jmlr.org/papers/v21/20-074.html>
25. 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 (2022)
26. Stanisławek, Tomasz: Kleister: key information extraction datasets involving long documents with complex layouts. In: Lladós, Josep, Lopresti, Daniel, Uchida, Seiichi (eds.) ICDAR 2021. LNCS, vol. 12821, pp. 564–579. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86549-8_36
27. Suadaa, L.H., Kamigaito, H., Funakoshi, K., Okumura, M., Takamura, H.: Towards table-to-text generation with numerical reasoning. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, vol. 1 (Long Papers), pp. 1451–1465. Association for Computational Linguistics (2021). <https://doi.org/10.18653/v1/2021.acl-long.115>. <https://aclanthology.org/2021.acl-long.115>

28. Vaswani, A., et al.: Attention is all you need. In: Proceedings of the 31st International Conference on Neural Information Processing Systems, pp. 6000–6010. NIPS 2017, Curran Associates Inc., Red Hook, NY, USA (2017)
29. Wiseman, S., Shieber, S., Rush, A.: Challenges in data-to-document generation. In: Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, pp. 2253–2263. Association for Computational Linguistics, Copenhagen, Denmark (2017). <https://doi.org/10.18653/v1/D17-1239>. <https://aclanthology.org/D17-1239>
30. Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M.: LayoutLM: pre-training of text and layout for document image understanding. In: Gupta, R., Liu, Y., Tang, J., Prakash, B.A. (eds.) KDD 2020: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, 23–27 August 2020, pp. 1192–1200. ACM (2020). <https://dl.acm.org/doi/10.1145/3394486.3403172>

2D-STRUCTURED LANGUAGE MODELING



CCpdf: Building a High Quality Corpus for Visually Rich Documents from Web Crawl Data

Michał Turski^{1,2}(✉), Tomasz Stanisławek¹, Karol Kaczmarek^{1,2}, Paweł Dyda^{1,2}, and Filip Graliński^{1,2}

¹ Snowflake, Warsaw, Poland

² Adam Mickiewicz University, Poznań, Poland

{michal.turski,tomasz.stanislawek,karol.kaczmarek,
pawel.dyda,filip.gralinski}@snowflake.com

Abstract. In recent years, the field of document understanding has progressed a lot. A significant part of this progress has been possible thanks to the use of language models pretrained on large amounts of documents. However, pretraining corpora used in the domain of document understanding are single domain, monolingual, or nonpublic. Our goal in this paper is to propose an efficient pipeline for creating a big-scale, diverse, multilingual corpus of PDF files from all over the Internet using Common Crawl, as PDF files are the most canonical types of documents as considered in document understanding. We analyzed extensively all of the steps of the pipeline and proposed a solution which is a trade-off between data quality and processing time. We also share a CCpdf corpus in a form of an index of PDF files along with a script for downloading them, which produces a collection useful for language model pretraining. The dataset and tools published with this paper offer researchers the opportunity to develop even better multilingual language models.

Keywords: Natural Language Processing · language models · dataset construction · document understanding

1 Introduction

Natural Language Processing (NLP) in recent years has made significant progress thanks to using language models such as GPT-3 [6] or T5 [23]. Usually these models are trained in a two-step process. The first part is pretraining, which utilizes a large corpus of text, and the second step is finetuning on a final task. Recent works demonstrate a considerable impact of pretraining on the final performance of a model [10, 17, 27]. For instance, GPT-3 was pretrained on a combination of texts from Common Crawl, WebText2, two book corpora, and the English Wikipedia (499 billion tokens in total) [6] while T5 was pretrained on the C4 corpus, which is 750 GB of data [23].

Work done while at Applica.ai, later acquired by Snowflake.

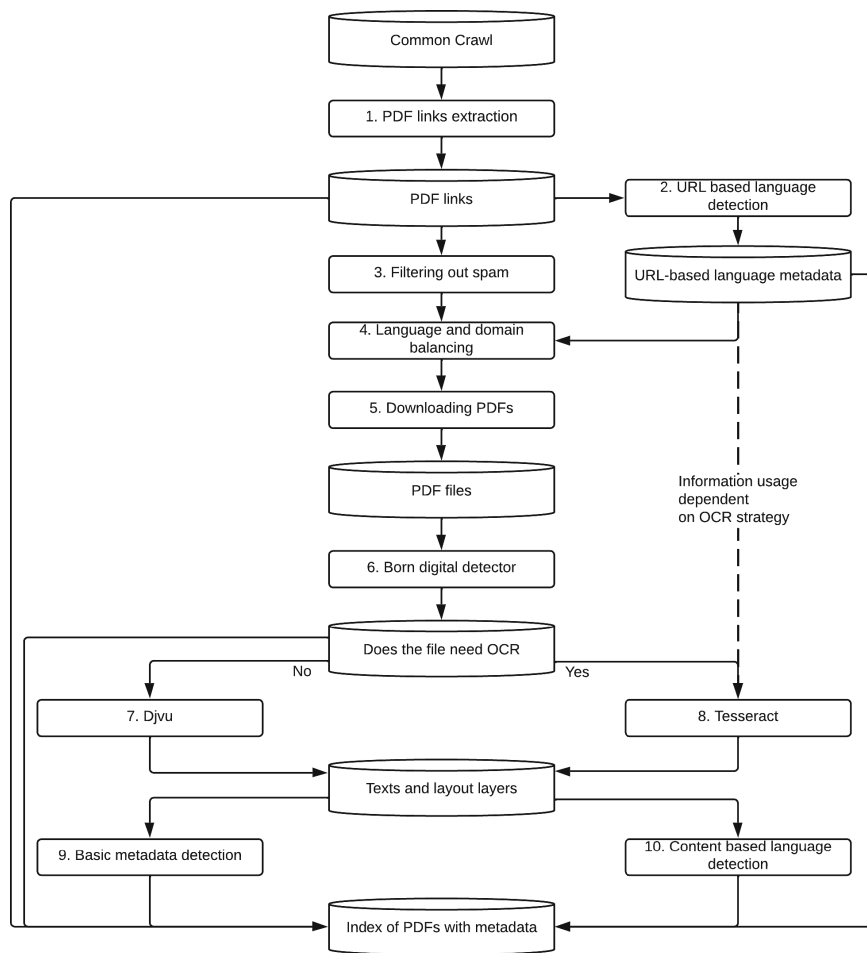


Fig. 1. The full flow of the process. Cylinders represent data, rectangles represent processing steps, and arrows represent data flow. A solid line indicates that the information is always used, and a dashed line represents data usage dependent on the processing strategy.

The recent progress in document understanding (defined as “capacity to convert a document into meaningful information” [5]) has been possible thanks to 2D language models such as LayoutLM [29,30], LAMBERT [9], or TILT [21]. Similarly to the models mentioned above, they also need large amounts of data for pretraining. The input to these models is a multi-modal representation of a document, e.g. tokens with their positions and images of pages.

The World Wide Web abounds in multi-modal documents, which contain enormous amounts of information. This information can be used in multiple domains: NLP, law, knowledge extraction, history, and many more. Yet, this aspect of the Internet remains relatively unexplored. So far, attempts of document dataset creation have been focused on either single domain (e.g. medical¹,

¹ <https://www.ncbi.nlm.nih.gov/pmc/tools/openftlist/>.

academic [3], or industrial [14]), while there has been no all-over-the-Internet approach. On the other hand, existing all-over-the Internet corpora (e.g. Web 1T 5-gram²) were focused on text only, not on multi-modal documents.

A document is a multi-modal form of communication: to interpret documents properly, we have to understand not only text, but also the layout and graphical elements. The most popular and portable multi-modal document format is PDF. In this study, we aim to describe a carefully designed pipeline for PDF corpus creation. We investigated numerous possible processing techniques and described their impact on the final data, which allowed us to achieve satisfactory trade-off between data quality, computing time, and monetary cost. The dataset itself (in the form of an index of PDF files and a script for downloading them) is also available at our website³. We share a corpus of 14.5M pages. It is useful as a dataset for 2D language model pretraining, but may also be employed as a source for derived datasets, in the same way as the IIT-CDIP dataset [14] was used to create many diverse challenges. Finally, analysis of the collected PDF files themselves yields helpful insight for language model creators, but also enhances our understanding of the World Wide Web as a source of PDF documents.

2 Related Works

The general problem of creating a large-scale corpus of documents has been studied extensively in recent years. IIT-CDIP [14] is a 40M pages (but according to the authors of OCR-IDL [4] only 35.5M of them are still reachable) dataset of reports from the Legacy Tobacco Documents Library⁴ collection, which was later reused to prepare a 400k page document classification dataset [12]. Also, OCR-IDL [4] reused IIT-CDIP to publish a 26M page dataset with high-quality OCR output. DoRe [19] is a French dataset of 2350 annual reports from 336 companies, unfortunately the data weren't shared publicly. There are also two layout analysis datasets based on scientific articles: Docbank [15] and Publaynet [33]. Their volumes are 500k and 360k pages, respectively. In addition, Ammar *et al.* [3] provided corpora of scientific documents together with a literature graph (defined as “a directed property graph which summarizes key information in the literature and can be used to answer the queries mentioned earlier as well as more complex queries” [3]). The National Library of Medicine has shared a PMC Open Access Subset⁵ which is a corpus of open-access, open-licensed medical publications. Allison *et al.* [2] proposed a pipeline for creating a corpus of PDFs sourced from the Internet. The goal of this work is to “identify key edge cases or common deviations from the format’s specification”. They also provide analyses of files in their corpus. All of these datasets are single-domain or single-language collections (usually both), while our aim is to create a diverse, multilingual dataset. There exists only one publication presenting such a dataset [31], but the authors limited

² <https://catalog.ldc.upenn.edu/LDC2006T13>.

³ <https://github.com/applicaai/CCpdf>.

⁴ <https://industrydocuments.ucsf.edu/tobacco/>.

⁵ <https://ncbi.nlm.nih.gov/pmc/tools/openftlist/>.

themselves to describing the data processing pipeline without analyzing their decisions. Also, their dataset was not shared.

Attempts have also been undertaken to create diversified corpora of texts sourced from the Internet. For instance, in CCNet [28], Common Crawl was used to create curated monolingual corpora in more than 100 languages. Also Schwenk *et al.* used Common Crawl in CCMatrix [24], but their purpose was to extract parallel sentences in different languages. The result was 10.8 billion parallel sentences in 90 languages. Another study in this vein is Smith *et al.* [25], whose method allowed to extract a 278 million token corpus of parallel English-French, English-Spanish, and English-German texts. In CCQA [13], a method for composing multilingual question-answering task using Common Crawl was proposed. The authors shared 130 million question-answer pairs. Liu and Curran [16] used Open Directory Project⁶ to extract a topic-diverse English corpus of 10 billion words. To pretrain the T5 language model [23], the authors extracted a 750 GB English text corpus, called C4, employing Common Crawl. Dodge *et al.* [7] explored this dataset further and analyzed the effects of the applied filtering. A similar pipeline to that used for C4 was applied to create the mT5 [32] training corpus, which is a multilingual version of T5. The proposed corpus has 6.3 trillion tokens. Qi *et al.* [22] crawled 10 million images with captions from the Internet and used it to pretrain the multi-modal ImageBERT model. C4Corpus [11] (not to be confused with C4 proposed by Raffel *et al.*, described above) utilized Common Crawl resources to provide multilingual (more than 50 languages) over 10 billion token corpus to the community. The Pile [8] is a 885 GB text corpus composed of 22 different datasets, and one of its subparts are texts from Common Crawl. Abadji *et al.* [1] proposed a document-oriented multilingual 12 GB corpus of texts from Common Crawl with quality annotations. It must be noted that the authors define the term “document” as a long, coherent piece of text, not as a PDF file, as we do in this study. Luccioni and Viviano [18] analyzed Common Crawl in terms of undesirable content, including hate speech and sexually explicit content, and investigated different filtering methods.

Table 1. Comparison of existing publicly available corpora. *Numbers of valid documents/pages according to the authors of OCR-IDL [4].

Dataset	Documents	Pages	Avg pages per doc	Languages	Domains	Years
IIT-CDIP	6.5M*	35.5M*	5.5	1	Industry documents	1990s
OCR-IDL	4.6M	26M	5.7	1	Industry documents	1990s
CCpdf	1.1M	14.5M	12.9	11	Multi-domain	Mostly 2010–2022

⁶ <http://odp.org>.

3 Collecting and Processing PDFs

In this section we describe how we addressed the challenge of finding, downloading, and processing a great volume of PDF documents. The full process is presented in Fig. 1.

3.1 Common Crawl

As our input we used web indexes created by Common Crawl⁷. Common Crawl is a project of The Internet Archive⁸ – an organization dedicated to providing a copy of the Internet to the community. They crawl webpages and save them into crawls dumps. A crawl dump contains billions of webpages (hundreds of terabytes of uncompressed data) and a new dump has been published nearly every month since March 2014. Some earlier, more irregular dumps starting from 2008 are also available.⁹ Each dump also contains an index of the crawled pages.

We decided to simply use the latest (and the largest) dump available at the time of writing this paper - the May 2022 dump.¹⁰ It contains 3.45 billion web pages, which amounts to 462 TB of uncompressed content. It would obviously be possible to apply the extraction procedure described in this paper to all crawls to obtain an even larger collection of PDFs, which would also allow for a diachronic analysis, but we wanted to focus on the most recent documents.

Note that dumps contain only files considered as text files by the Common Crawl web robot. Mostly these are web pages in the HTML format, but, fortunately, PDFs are also treated as text files, being derivative of the PostScript page description language. This is not the case with, for instance, images, Excel files, DOCX files. Consequently, such files cannot be amassed using the methods described in the aforementioned papers.

3.2 PDF Links Extraction

We experimented with two methods for extracting links to PDF files (step 1 in Fig. 1):

1. using CDX files, i.e., index server files provided by Common Crawl;
2. looking for links to PDF files in WARC, i.e., raw crawl data files.

The first method is simpler, as CDX files are easy to download and take up only 225 GB in total. The second method might yield more links to PDF files, but:

- it is impossible for us to download all WARCs. Only a limited number of them can be processed, though still a significant number of PDF links can be added even if a small percentage of all WARC files are processed,

⁷ <https://commoncrawl.org>.

⁸ <https://archive.org/>.

⁹ <https://commoncrawl.org/the-data/get-started/>.

¹⁰ <https://commoncrawl.org/2022/06/may-2022-crawl-archive-now-available/>.

- there is lower probability that the file linked is available at all, be it in the crawl dump or simply at the original address.

In CDX files, the MIME type of a captured file is specified, and we limited ourselves to the `application/pdf` type.

Hence, in this paper, we focus on the first method, which allows to speed up the whole processing pipeline.

3.3 URL-Based Language Detection

We decided to limit our investigation to the following set of 11 languages: Arabic, Dutch, English, French, German, Italian, Japanese, Polish, Portuguese, Russian, and Spanish.

When deciding whether to process a given URL, we applied a number of simple heuristics to determine the language. For example, we assumed that PDFs from `.pl` domains are Polish unless there is `lang=en` inside the URL etc. Note that this is a preliminary filter; later, when the contents have been downloaded, we do a proper language detection (see Sect. 3.9).

In August 2018, Common Crawl added language metadata to CDX files.¹¹ Unfortunately, the Compact Language Detector 2 employed there is applicable only for plain texts or HTMLs, and only a small percentage of PDF links contained the language metadata; therefore, it was unusable for our purposes.

This step of the pipeline is presented as block 2 in Fig. 1.

3.4 Filtering Out Spam

One of the challenges to be tackled in Web information retrieval or when creating a massive text corpus sourced from the Web is the problem of (web) spam and, more generally, low quality pages (step 3 in Fig. 1). Web spam is usually related to black-hat search engine optimization, i.e., creating link farms of web pages with automatically or semi-automatically generated content. It turns out that PDF files found on the Internet have the advantage of a relatively low percentage of spam, especially when compared to HTML web pages. More generally, we believe PDF files usually contain more formal content as most of them are business, legal, or scientific documents.

Still, some spam PDFs were found in Common Crawl dumps. Fortunately, the way in which spammers operate is rather homogeneous. A typical telltale of a spammy PDF was a long name composed of lower-case letters interspersed with hyphens. A regular expression was written to detect suspicious URLs, and if a domain happened to contain a large percentage of such URLs, it was assumed to be spammy as a whole and totally discarded. Thanks to this simple heuristic, in a sample of 1k documents we manually annotated (see Sect. 3.9) we found no spam PDFs.

¹¹ <https://commoncrawl.org/2018/08/august-2018-crawl-archive-now-available/>.

3.5 PDF Data Download Methods

In order to ensure diversity, we downloaded at most three PDF files from each domain for a language in a random but reproducible manner. For English and German this number was lowered to, respectively, one and two, as PDFs in these two languages are much more numerous compared to others. This limitation also serves as a filter against anomalies such as millions of PDFs coming from a single domain; especially a spammy one, if not detected with the procedure described in Sect. 3.4. Balancing is represented as step 4 in Fig. 1.

The files were downloaded from the original URLs (step 5 in Fig. 1). Optionally, one could extract the file from a Common Crawl dump, especially if the file is not available at the original site. We provide a script to extract PDF files directly from the dump; fortunately, one does not need to download the whole dump to extract a file.

There is, however, one serious issue with extracting PDFs from crawl dumps: all files are truncated by the crawler to 1 MB. This limit is quite high for HTML pages, but unfortunately rather low for PDF files. This means that only small-sized PDF files can be extracted from Common Crawl dumps; larger ones have to be downloaded from the original sites.

The final and intermediary statistics for the files downloaded are presented in Table 2.

Table 2. Number of documents per processing step and language. Percentage values show success rates of downloading (in the downloaded column) or downloading and processing together (in the processed column). The success rate for processing a downloaded document equals 94.94%.

	URLs found	Anti-spam filtered	Domain balanced	Language balanced	Successfully downloaded	Successfully processed
ar	65395	65374	13142	13142	11 710 (89.10%)	10 826 (82.38%)
de	1661317	1659713	320978	200000	182 607 (91.30%)	172 668 (86.33%)
en	11515766	11501781	952776	200000	182 071 (91.04%)	175 440 (87.72%)
es	871843	871478	106143	106143	93 163 (87.77%)	88 952 (83.80%)
fr	654250	653120	143020	143020	129 927 (90.85%)	121 905 (85.24%)
it	831344	831026	129610	129610	119 731 (92.38%)	114 265 (88.16%)
ja	1160543	1160410	151686	151686	139 990 (92.29%)	134 310 (88.54%)
nl	339519	338946	92372	92372	84 848 (91.85%)	79 720 (86.30%)
pl	438770	438531	85635	85635	79 668 (93.03%)	75 374 (88.02%)
pt	697535	697285	73130	73130	64 725 (88.51%)	61 405 (83.97%)
ru	628473	628061	105535	105535	91 708 (86.90%)	85 552 (81.07%)
all	18864755	18845725	2174027	1300273	1 180 148 (90.76%)	1 120 417 (86.17%)

3.6 Born Digital Scanner

To process correctly all kinds of documents in the document understanding domain we need to extract tokens from PDF files with their bounding boxes sorted properly, i.e., according to the reading order. The most common approach

[9, 21, 29, 30] is to process each PDF file with the use of some OCR engine, e.g. Tesseract [26], Amazon Textract¹², Microsoft Azure Computer Vision API,¹³ or Google Vision API¹⁴. This method simplifies the processing pipeline and removes the need to understand the complicated PDF file format.

The biggest challenge in direct text and layout extraction lies in processing image content since there is no easy way to detect whether an image contains text. On the other hand, some documents lack pictures altogether; instead they contain textual information in the PDF file structure. We call them *documents that do not require OCR*. From such documents text can be extracted along with bounding boxes using dedicated Python libraries, such as pdfminer.six¹⁵, pdfplumber,¹⁶ or a DjVu-based tool¹⁷. Direct text extraction using these tools leads to the reduction of the processing time and improvement of the quality of the extracted data by preventing OCR errors. Therefore, we decided to introduce a mechanism, called the Born Digital detector, for finding these kinds of documents (step 6 in Fig. 1).

3.7 Born Digital Detection Heuristics

In order to detect documents that do not need to be processed with an OCR pipeline, we created a fast, simple heuristic-based classifier:

- *Visible Text Length* > 100 - Visible text in the document contains more than 100 characters
- *Hidden Text Length* = 0 - There is no hidden text in the document
- *Image Count* = 0 - There are no images in the document

Used statistics (*Visible Text Length*, *Hidden Text Length*, *Image Count*) were extracted using *Digital-born PDF Scanner*¹⁸ tool written by us.

Our simple method was able to classify 219 documents out of 967 as born-digital files that do not require OCR. (In other words, we can skip the time-consuming OCR process for more than 1 out of 5 PDF files). To check quality of our heuristic we manually annotated the same sample of documents. The precision of the proposed method was 93.15%. All errors (15) were caused by adding a background with logo text to the file. In the future, we can also improve that kind of cases by extracting metadata information about PDF file background as well.

¹² <https://aws.amazon.com/textract/>.

¹³ <https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/overview-ocr>.

¹⁴ <https://cloud.google.com/vision/docs/pdf>.

¹⁵ <https://github.com/pdfminer/pdfminer.six>.

¹⁶ <https://github.com/jsvine/pdfplumber>.

¹⁷ <http://jwilk.net/software/pdf2djvu>, <https://github.com/jwilk/ocrodjvu>.

¹⁸ <https://github.com/applicaai/digital-born-pdf-scanner>.

Table 3. Results for the Born Digital detector mechanism.

	Gold standard #	Born Digital detector			
		Precision	Recall	F1-score	TP + FP #
born digital, OCR not required	471 (48.71%)	93.15	43.31	59.13	219 (22.65%)
born digital, OCR required	321 (33.20%)	–	–	–	–
scan	175 (18.10%)	–	–	–	–
all	967	–	–	–	–

3.8 OCR Processing

One of the initial steps of the PDF processing pipeline is the URL based language detection method (see Sect. 3.3). Information about the language of the document is needed for filtering documents for specific languages and also by the OCR tool. In the next step (see Sect. 3.7), we select PDF files for processing either by the DjVu-based tool (if the file is born digital then it does not require OCR) or by Tesseract OCR [26]. The result is hocr files containing extracted text with its bounding boxes. This form of data serves as the input to the subsequent processing and analyzing steps.

Table 4. Comparison of resource utilization for different strategies of the text extraction from PDF files. *for Azure OCR we used 4 CPU (which is minimal recommendation for container in version 3.2) and multiplied the number by 4.

Strategy name	Processing time (using 1 CPU)		Additional cost	
	1k files	in relation to fastest	Single page	1k files
DjVu-based tool + Born-digital detector	5.6 h	1x	-	-
Tesseract + URL based LD	23.7 h	4x	-	-
Tesseract + Build-in LD mechanism	75.9 h	14x	-	-
MSOCR + Build-in LD mechanism	16.7 h*	3x	0.001\$	13\$

Possible Alternatives. In a typical scenario of extracting text with bounding boxes from a PDF file, researchers use a custom OCR engine [9, 21, 29, 30], e.g. Tesseract, Microsoft Azure Computer Vision API, Amazon Textract, or Google Vision API. However, when we want to process millions of PDF files, we need to think about the utilization of resources in the context of time and money. Additionally, contrary to previous work, the language of a PDF file that we want to process is unknown. Therefore, to choose the most economical option, we tested the following strategies:

1. *DjVu-based tool with a born-digital detector* – for details, please see Sect. 3.7
2. *Tesseract with URL based Language Detection (LD)* – described at the beginning of this section
3. *Tesseract with a built-in LD mechanism* – in this strategy, we use the Tesseract OCR [26] engine with a built-in language detection mechanism

4. *Azure CV API with a built-in LD mechanism* – in this strategy we use Microsoft Azure Computer Vision API¹⁹ with a built-in language detection mechanism

We achieved the shortest processing time (see Table 4) with the DjVu-based tool and a born-digital detector (see Sect. 3.7), which followed from the fact that we did not need to run any ML models. Also quality of output from the DjVu-based tool is better than from any OCR engine, because it extracts real content of a file and does not introduce any processing noise. *Azure CV API* and *Tesseract with URL based language detection* are the slowest OCR engines with 3-4 longer processing time. It's turn out that the slowest processing time has strategy is *Tesseract with build-in LD mechanism* and, therefore, we will not apply it in our final pipeline.

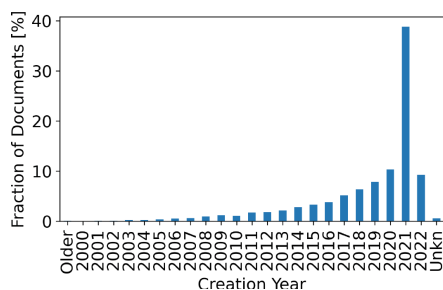


Fig. 2. Distribution of the analyzed sample in terms of creation year.

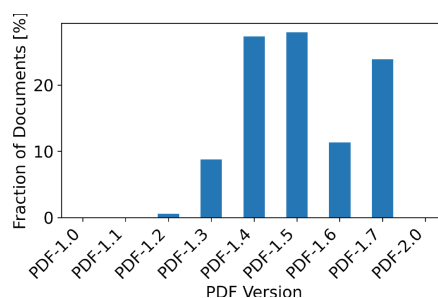


Fig. 3. Distribution of the analyzed sample according to PDF version.

3.9 Language Identification

In our final processing pipeline we used two language detection mechanisms:

1. URL-based method. Described in Sect. 3.3.
2. Content based method. We used the *langdetect*²⁰ library to detect language based on its text content extracted in the previous step.

We tested the quality of our language detection methods on ~1k manually annotated documents (Table 5). Both of our mechanisms can detect only a single language but, in reality, we found out that 27 documents had multiple languages (in 23 cases one of them was English). Fortunately, detecting a single language allowed us to predict the language correctly for almost all documents (97.3%).

With the use of the URL based method, we achieved a 90.51% F1-score on average, which seems reasonably good when we take into account the simplicity of the method. The content based method works better in general with an F1-score of 94.21% on average. The single exception here is the Japanese language. Both mechanisms produced the least satisfactory results for two languages: Arabic and English. It turned out that many documents from the *.ar* domain were actually in English. Therefore, for the content based mechanism we wrongly processed the PDF files with the Arabic Tesseract model.

¹⁹ <https://docs.microsoft.com/en-us/azure/cognitive-services/computer-vision/overview-ocr>.

²⁰ <https://pypi.org/project/langdetect/>.

Additionally, we found out that when we used the proper Tesseract model our results increased drastically to an F1-score of 98.05% on average. The main reason why this happened was the fact that the language identification mechanism was working on the proper alphabet.

Possible Alternatives. In Table 6 we present the results for different language identification tools. All of them achieved similar F1-scores, of which *spacy* (94.33%) and *langdetect* (94.21%) performed best. When we also take into consideration the processing time, it turns out that *gclid3* was the best one with a huge advantage over the second tool, which was the *langdetect* library. Therefore, we decided to balance quality and resource utilization and use *langdetect* as our main tool for language identification.

Table 5. Quality of the language identification methods verified on 996 manually annotated documents.

	Gold standard #	URL based method			Content based method					
					All documents			Proper Tesseract lang		
		Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1-score
ar	20	46.51	95.24	62.50	44.19	95.00	60.32	100.0	100.0	100.0
de	94	94.68	92.71	93.68	98.94	98.94	98.94	100.0	100.0	100.0
en	119	80.46	58.82	67.96	94.34	84.03	88.89	98.55	98.55	98.55
es	75	94.52	93.24	93.88	98.65	97.33	97.99	98.57	98.57	98.57
fr	108	93.94	86.92	90.29	100.0	91.67	95.65	100.0	100.0	100.0
it	101	93.20	95.05	94.12	98.97	95.05	96.97	94.79	94.79	94.79
jp	108	100.0	98.10	99.04	100.0	89.81	94.63	92.38	92.38	92.38
nl	90	84.91	100.0	91.84	98.86	96.67	97.75	96.67	96.67	96.67
pl	88	95.56	100.0	97.73	98.86	98.86	98.86	98.86	98.86	98.86
pt	83	94.38	98.82	96.55	97.62	98.80	98.21	98.78	98.78	98.78
ru	78	96.34	98.75	97.53	97.47	98.72	98.09	100.0	100.0	100.0
other	2	0	0	0	0	0	0	0	0	0
no text	3	0	0	0	0	0	0	0	0	0
multi	27	0	0	0	0	0	0	0	0	0
all	996	88.59	92.51	90.51	93.45	94.99	94.21	98.05	98.05	98.05

Table 6. Comparison of the quality and processing time of different language identification tools.

Tool name	F1-scores for content based method												Processing time	
	ar	de	en	es	fr	it	jp	nl	pl	pt	ru	all	1k files	1M files
langdetect ^a	60.32	98.94	88.89	97.99	95.65	96.97	94.63	97.75	98.86	98.21	98.09	94.21	0.28 min	4.67 h
lingua-py ^b	60.32	97.90	88.79	96.69	94.74	95.92	98.59	96.77	99.44	98.18	97.47	94.05	2.57 min	42.8 h
spacy ^c	60.32	98.94	87.33	97.99	94.79	97.49	98.59	97.18	100.0	98.18	97.47	94.33	3.62 min	60.3 h
gclid3 ^d	59.01	98.94	89.91	97.96	96.15	97.49	92.16	97.73	99.44	98.78	98.07	94.08	0.03 min	0.33 h

^a <https://pypi.org/project/langdetect/>

^b <https://github.com/pemistahl/lingua-py>

^c https://spacy.io/universe/project/spacy_fastlang

^d <https://pypi.org/project/gclid3/>

3.10 Produced Index

As a result of our pipeline, we created an index of successfully downloaded and processed files. We decided to download up to 200k documents per language to share a reasonably sized corpus, with a good diversity of languages. It gives an acceptably good trade-off between the balance of languages and the size of the dataset. Statistics about the index are presented in Table 2.

A comparison of our dataset to existing corpora is presented in Table 1. The corpus we provided is smaller than the previous ones considering the total number of documents and pages. Still, language models will benefit in many aspects, (1) understanding long-distance relationships as the dataset has, on average, the longest documents compared to previous works, (2) multi-language training as we selected 11 different languages, (3) multi-domain training as we sourced documents from different websites all over the Internet, (4) document understanding of recently created documents (which may differ from the old ones in terms of language, layout, and graphical style) as the majority of files in our corpus were produced after 2010 (in IIT-CDIP, the most popular corpus so far, all the documents were created in the 90s).

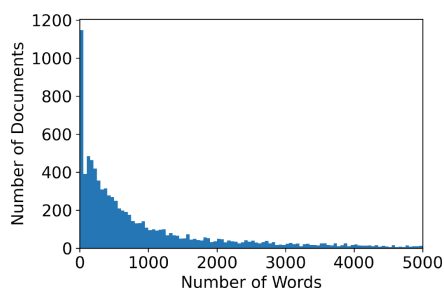


Fig. 4. Distribution of word count per document.

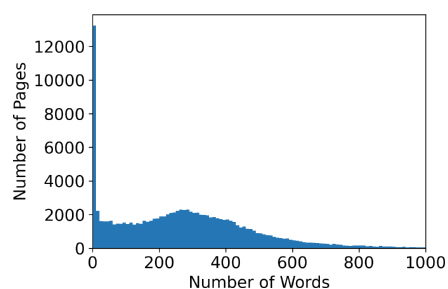


Fig. 5. Distribution of word count per page.

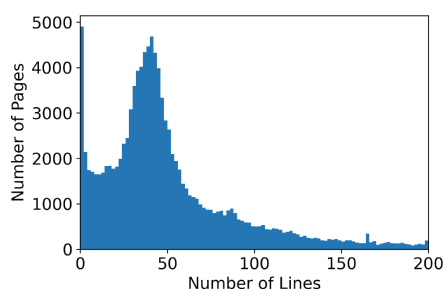


Fig. 6. Distribution of the number of lines per page.

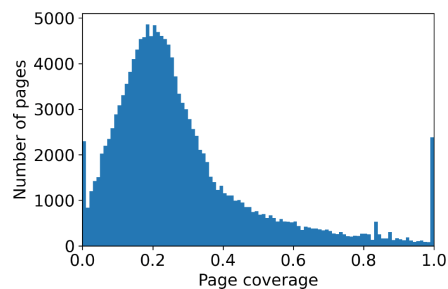


Fig. 7. Distribution of text coverage of page.

4 Exploration of PDFs

Since we provide a large scale, highly diversified collection of PDFs downloaded from all over the Internet, we want to provide some insight into the properties

graphics. The typical value of words per page is between 0 and 500, and the typical value of lines per page is between 0 and 55.

To provide some insight into the layout of the documents, we checked to what extent each page was covered by bounding boxes of tokens. We may look at it as part of the text coverage parameter. Distribution of this value is presented in Fig. 7. 76.2% of our sample fell into the range of 5% to 40% with respect to that parameter. Similarly to the previously described properties, once again we see a peak for empty pages. There is also a peak for pages fully or almost fully covered by text.

We were also interested in the ratio of page dimensions. 99.7% of x/y ratios were in the range of 0.4 to 2; the smallest value being 0.09, and the largest – 4.79. In our sample, 65.0% were pages with the dimension ratio close to $\sqrt{2}$, which is a standard ratio for the A, B and C paper series. 86.9% of them were vertical pages, and 13.1% – horizontal ones. Also, the LETTER format was popular; it comprised 10.6% of the sample: 92.9% documents were vertical, and 7.1% horizontal. In total, the A, B, C, and LETTER series comprised 75.5% of the sample.

To gather more information about the layout of the documents, we created heatmaps of token bounding boxes for vertical and horizontal pages (Figs. 9 and 10, respectively). As we can see, layouts with two columns of text are fairly popular, especially for horizontal pages. Also, text occurs more frequently on the right side of a page.

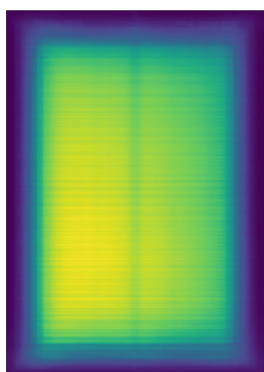


Fig. 9. Heatmap for vertical pages (brighter means more tokens, darker – fewer tokens).

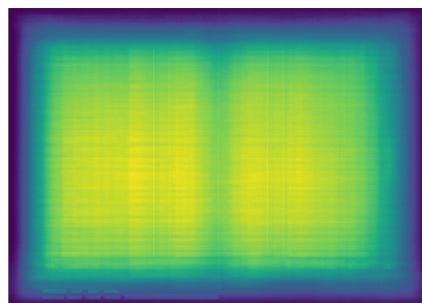


Fig. 10. Heatmap for horizontal pages (brighter means more tokens, darker – fewer tokens).

5 Discussion

In this study we analyzed a pipeline for creating a corpus of documents. According to our experiments, the most effective way of OCR processing of PDF files is a two-step procedure. The first step consists in the classification of the files according to whether they need an OCR engine or simple text extraction is sufficient. In the second step, we process the file with either an OCR engine (in our

case Tesseract) or an extraction tool (in our case the DjVu-based tool). In the former scenario, we also discovered that predefining the OCR language speed up the process substantially; unfortunately, this comes at some cost in terms of data quality. However, this cost may be mitigated by a simple heuristic which filters out documents where the predefined OCR language did not match the one discovered by the language detector. We also analyzed different language detection tools in terms of output quality and processing time, and discovered that the langdetect tool offered the best trade-off between these values.

One of the limitations of this research study was that we focused only on the processing pipeline without analyzing the impact of each project decision on the final language model. However, this kind of study would be very expensive, as it would require multiple pretrainings of a language model. Language model pretraining is a costly process in terms of money, time, and environmental impact [20].

Also, conclusions drawn from the analysis of our sample can hardly be generalized to the whole content of the Internet and only provide some insight, rather undisputed knowledge. This follows from the filtering procedure: we decided to down-sample document-rich domains and languages, therefore, statistics calculated on the whole content of the Internet may differ from the ones presented in this work.

The approach which we used to create the dataset may be reused to all of the previous Common Crawl dumps in the WARC format, of which there are 84 in total. We decided to limit ourselves to one dump only due to computational and storage limitations. One with enough computing resources may easily reproduce our pipeline and create a corpus up to 84 times larger.

6 Conclusions

Large corpora of documents are crucial for 2D language model pretraining. Recent approaches to their creation have had limitations in terms of diversity and multilinguality. Diversity of the dataset is a crucial property, as data used in the training phase impact the biases of the model. Efficient design of a pipeline for creating such a corpus has not been studied before. In this work we addressed those limitations by designing a process of downloading diversified samples of PDFs and their efficient processing. To obtain documents we used Common Crawl, which is a popular source of data for language model pretraining, but has rarely been used in the context of 2D language models. The PDF files used for this project were balanced across languages and domains, which guarantees diversity with respect to layouts and topics. To make the processing pipeline efficient in terms of computing time and data quality, we tested different strategies of OCR processing, i.e. usage of the embedded textual layer for documents not requiring OCR, and predefining the OCR language. The language detection step was also carefully analyzed.

The result of this work is an index of PDF files with their URL addresses and metadata, and the script for downloading it is available at our repository²¹. The supplied data were analyzed in terms of not only document length and layout, but also metadata connected to the PDF format (i.e., the PDF version and the creator tool), which can help understand better the dataset itself, but also give an insight into the content of the Internet.

Acknowledgments. The Smart Growth Operational Programme partially supported this research under projects no. POIR.01.01.01-00-0877/19-00 (*A universal platform for robotic automation of processes requiring text comprehension, with a unique level of implementation and service automation*) and POIR.01.01.01-00-1624/20 (*Hiper-OCR - an innovative solution for information extraction from scanned documents*).

References

1. Abadji, J., Suarez, P.O., Romary, L., Sagot, B.: Towards a cleaner document-oriented multilingual crawled corpus. ArXiv abs/2201.06642 (2022)
2. Allison, T., et al.: Research report: Building a wide reach corpus for secure parser development. In: 2020 IEEE Security and Privacy Workshops (SPW), pp. 318–326 (2020). <https://doi.org/10.1109/SPW50608.2020.00066>
3. Ammar, W., et al.: Construction of the literature graph in semantic scholar. In: Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 3 (Industry Papers), pp. 84–91. Association for Computational Linguistics, New Orleans - Louisiana (2018). <https://doi.org/10.18653/v1/N18-3011>. <https://aclanthology.org/N18-3011>
4. Biten, A.F., Tito, R., Gomez, L., Valveny, E., Karatzas, D.: OCR-IDR: OCR annotations for industry document library dataset. arXiv preprint [arXiv:2202.12985](https://arxiv.org/abs/2202.12985) (2022)
5. Borchmann, L., et al.: DUE: end-to-end document understanding benchmark. In: NeurIPS Datasets and Benchmarks (2021)
6. Brown, T., et al.: Language models are few-shot learners. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M., Lin, H. (eds.) Advances in Neural Information Processing Systems. vol. 33, pp. 1877–1901. Curran Associates, Inc. (2020). <https://proceedings.neurips.cc/paper/2020/file/1457c0d6bfc4967418bfb8ac142f64a-Paper.pdf>
7. Dodge, J., et al.: Documenting large webtext corpora: a case study on the colossal clean crawled corpus. In: Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pp. 1286–1305. Association for Computational Linguistics, Online and Punta Cana, Dominican Republic (2021). <https://doi.org/10.18653/v1/2021.emnlp-main.98>. <https://aclanthology.org/2021.emnlp-main.98>
8. Gao, L., et al.: The Pile: an 800gb dataset of diverse text for language modeling. arXiv preprint [arXiv:2101.00027](https://arxiv.org/abs/2101.00027) (2020)
9. Garncares, L., et al.: LAMBERT: layout-aware language modeling for information extraction. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12821, pp. 532–547. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86549-8_34








²¹ <https://github.com/applicaai/CCpdf>.

10. Gururangan, S., et al.: Don't stop pretraining: adapt language models to domains and tasks. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.acl-main.740>. <http://dx.doi.org/10.18653/v1/2020.acl-main.740>
11. Habernal, I., Zayed, O., Gurevych, I.: C4Corpus: Multilingual web-size corpus with free license. In: Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC2016), pp. 914–922. European Language Resources Association (ELRA), Portorož, Slovenia (2016). <https://aclanthology.org/L16-1146>
12. Harley, A.W., Ufkes, A., Derpanis, K.G.: Evaluation of deep convolutional nets for document image classification and retrieval. In: ICDAR (2015)
13. Huber, P., et al.: CCQA: a new web-scale question answering dataset for model pre-training (2021)
14. Lewis, D., Agam, G., Argamon, S., Frieder, O., Grossman, D., Heard, J.: Building a test collection for complex document information processing. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 665–666. SIGIR 2006, Association for Computing Machinery, New York, NY, USA (2006). <https://doi.org/10.1145/1148170.1148307>
15. Li, M., et al.: DocBank: a benchmark dataset for document layout analysis (2020)
16. Liu, V., Curran, J.R.: Web text corpus for natural language processing. In: 11th Conference of the European Chapter of the Association for Computational Linguistics. Association for Computational Linguistics, Trento, Italy (2006). <https://www.aclweb.org/anthology/E06-1030>
17. Liu, Y., et al.: RoBERTa: a robustly optimized BERT pretraining approach (2019)
18. Luccioni, A.S., Viviano, J.D.: What's in the box? An analysis of undesirable content in the Common Crawl corpus. In: ACL (2021)
19. Masson, C., Paroubek, P.: NLP analytics in finance with DoRe: a French 250M tokens corpus of corporate annual reports. In: Proceedings of The 12th Language Resources and Evaluation Conference, pp. 2261–2267. European Language Resources Association, Marseille, France (2020). <https://www.aclweb.org/anthology/2020.lrec-1.275>
20. Patterson, D.A., et al.: Carbon emissions and large neural network training. ArXiv abs/2104.10350 (2021)
21. Powalski, R., Borchmann, Ł, Jurkiewicz, D., Dwojak, T., Pietruszka, M., Pałka, G.: Going full-TILT boogie on document understanding with text-image-layout transformer. In: Lladós, J., Lopresti, D., Uchida, S. (eds.) ICDAR 2021. LNCS, vol. 12822, pp. 732–747. Springer, Cham (2021). https://doi.org/10.1007/978-3-030-86331-9_47
22. Qi, D., Su, L., Song, J., Cui, E., Bharti, T., Sacheti, A.: ImageBERT: cross-modal pre-training with large-scale weak-supervised image-text data (2020)
23. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer (2019)
24. Schwenk, H., Wenzek, G., Edunov, S., Grave, E., Joulin, A.: CCMatrix: mining billions of high-quality parallel sentences on the web. In: ACL (2021)
25. Smith, J.R., Saint-Amand, H., Plamada, M., Koehn, P., Callison-Burch, C., Lopez, A.: Dirt cheap web-scale parallel text from the common Crawl. In: Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), pp. 1374–1383. Association for Computational Linguistics, Sofia, Bulgaria (2013). <https://www.aclweb.org/anthology/P13-1135>
26. Smith, R.: Tesseract open source OCR engine (2022). <https://github.com/tesseract-ocr/tesseract>

27. Turc, I., Chang, M.W., Lee, K., Toutanova, K.: Well-read students learn better: On the importance of pre-training compact models. arXiv preprint [arXiv:1908.08962v2](https://arxiv.org/abs/1908.08962v2) (2019)
28. Wenzek, G., et al.: CCNet: extracting high quality monolingual datasets from web crawl data (2019)
29. Xu, Y., et al.: LayoutLMv2: multi-modal pre-training for visually-rich document understanding. In: ACL-IJCNLP 2021 (2021)
30. Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M.: LayoutLM: pre-training of text and layout for document image understanding. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (2020)
31. Xu, Y., et al.: LayoutXLM: multimodal pre-training for multilingual visually-rich document understanding. arXiv preprint [arXiv:2004.21040](https://arxiv.org/abs/2004.21040) (2021)
32. Xue, L., et al.: mT5: a massively multilingual pre-trained text-to-text transformer. In: NAACL (2021)
33. 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). <https://doi.org/10.1109/ICDAR.2019.00166>



LAMBERT: Layout-Aware Language Modeling for Information Extraction

Lukasz Garncarek¹ , Rafał Powalski¹ , Tomasz Stanisławek^{1,2} ,
Bartosz Topolski¹ , Piotr Halama¹ , Michał Turski^{1,3} ,
and Filip Graliński^{1,3} 

¹ Applica.ai, Zajęcza 15, 00-351 Warsaw, Poland

{lukasz.garncarek, rafal.powalski, tomasz.stanislawek,
bartosz.topolski, piotr.halama, michal.turski, filip.gralinski}@applica.ai

² Warsaw University of Technology, Koszykowa 75, 00-662 Warsaw, Poland

³ Adam Mickiewicz University, 1 Wieniawskiego, 61-712 Poznań, Poland

Abstract. We introduce a simple new approach to the problem of understanding documents where non-trivial layout influences the local semantics. To this end, we modify the Transformer encoder architecture in a way that allows it to use layout features obtained from an OCR system, without the need to re-learn language semantics from scratch. We only augment the input of the model with the coordinates of token bounding boxes, avoiding, in this way, the use of raw images. This leads to a layout-aware language model which can then be fine-tuned on downstream tasks.

The model is evaluated on an end-to-end information extraction task using four publicly available datasets: Kleister NDA, Kleister Charity, SROIE and CORD. We show that our model achieves superior performance on datasets consisting of visually rich documents, while also outperforming the baseline RoBERTa on documents with flat layout (NDA F_1 increase from 78.50 to 80.42). Our solution ranked first on the public leaderboard for the Key Information Extraction from the SROIE dataset, improving the SOTA F_1 -score from 97.81 to 98.17.

Keywords: Language model · Layout · Key information extraction · Transformer · Visually rich document · Document understanding

1 Introduction

The sequential structure of text leads to it being treated as a sequence of tokens, characters, or more recently, subword units. In many problems related to Natural Language Processing (NLP), this linear perspective was enough to enable significant breakthroughs, such as the introduction of the neural Transformer architecture [28]. In this setting, the task of computing token embeddings is

L. Garncarek, R. Powalski, T. Stanisławek and B. Topolski—Equally contributed to the paper.

solved by Transformer encoders, such as BERT [6] and its derivatives, achieving top scores on the GLUE benchmark [29].

They all deal with problems arising in texts defined as sequences of words. However, in many cases there is a structure more intricate than just a linear ordering of tokens. Take, for instance, printed or richly-formatted documents, where the relative positions of tokens contained in tables, spacing between paragraphs, or different styles of headers, all carry useful information. After all, the goal of endowing texts with layout and formatting is to improve readability.

In this article we present one of the first attempts to enrich the state-of-the-art methods of NLP with layout understanding mechanisms, contemporaneous with [32], to which we compare our model. Our approach injects the layout information into a pretrained instance of RoBERTa. We fine-tune the augmented model on a dataset consisting of documents with non-trivial layout.

We evaluate our model on the end-to-end information extraction task, where the training set consists of documents and the target values of the properties to be extracted, without any additional annotations specifying the locations where the information on these properties can be found in the documents. We compare the results with a baseline RoBERTa model, which relies on the sequential order of tokens obtained from the OCR alone (and does not use the layout features), and with the solution of [31,32]. LAMBERT achieves superior performance on visually rich documents, without sacrificing results on more linear texts.

1.1 Related Work

There are two main lines of research into understanding documents with non-trivial layout. The first one is Document Layout Analysis (DLA), the goal of which is to identify contiguous blocks of text and other non-textual objects on the page and determine their function and order in the document. The obtained segmentation can be combined with the textual information contained in the detected blocks. This kind of method has recently been employed in [17].

Many services employ DLA functionality for OCR (which requires document segmentation), table detection or form field detection, and their capabilities are still expanding. The most notable examples are Amazon Textract [1], the Google Cloud Document Understanding AI platform [8], and Microsoft Cognitive Services [20]. However, each has limitations, such as the need to create rules for extracting information from the tables recognized by the system, or use training datasets with annotated document segments. More recent works on information extraction using DLA include, among others, [2,3,10,14,19,22,25]. They concentrate on specific types of documents, such as invoices or forms, where the layout plays a relatively greater role: more general documents may contain tables, but they can also have large amounts of unstructured text.

The second idea is to directly combine the methods of Computer Vision and NLP. This could be done, for instance, by representing a text-filled page as a multi-channel image, with channels corresponding to the features encoding the semantics of the underlying text, and, subsequently, using convolutional networks. This method was used, among others, by Chargrid and BERTgrid

models [5, 15]. On the other hand, LayoutLM [32] and TRIE [34] used the image recognition features of the page image itself. A more complex approach was taken by PICK [33], which separately processes the text and images of blocks identified in the document. In this way it computes the vertex embeddings of the block graph, which is then processed with a graph neural network.

Our idea is also related to the one used in [24], though in a different setting. They considered texts accompanied by audio-visual signal injected into a pretrained BERT instance, by combining it with the input embeddings.

LAMBERT has a different approach. It uses neither the raw document image, nor the block structure that has to be somehow inferred. It relies on the tokens and their bounding boxes alone, both of which are easily obtainable from any reasonable OCR system.

1.2 Contribution

Our main contribution is the introduction of a *Layout-Aware Language Model*, a general-purpose language model that views text not simply as a sequence of words, but as a collection of tokens on a two-dimensional page. As such it is able to process plain text documents, but also tables, headers, forms and various other visual elements. The implementation of the model is available at <https://github.com/applicaai/lambert>.

A key feature of this solution is that it retains the crucial trait of language models: the ability to learn in an unsupervised setting. This allows the exploitation of abundantly available unannotated public documents, and a transfer of the learned representations to downstream tasks. Another advantage is the simplicity of this approach, which requires only an augmentation of the input with token bounding boxes. In particular, no images are needed. This eliminates an important performance factor in industrial systems, where large volumes of documents have to be sent over a network between distributed processing services.

Another contribution of the paper is an extensive ablation study of the impact of augmenting RoBERTa with various types of additional positional embeddings on model performance on the SROIE [12], CORD [21], Kleister NDA and Kleister Charity datasets [27].

Finally, we created a new dataset for the unsupervised training of layout-aware language models. We will share a 200k document subset, amounting to 2M visually rich pages, accompanied by a dual classification of documents: business/legal documents with complex structure; and others. Due to IIT-CDIP Test Collection dataset [16] accessibility problems¹, this would constitute the largest widely available dataset for training layout-aware language models. It would allow researchers to compare the performance of their solutions not only on the same test sets, but also with the same training set. The dataset is published at <https://github.com/applicaai/lambert>, together with a more detailed description that is too long for this paper.

¹ The link <https://ir.nist.gov/cdip/> seems to be dead (access on Feb 17, 2021).

2 Proposed Method

We inject the layout information into the model in two ways. Firstly, we modify the input embeddings of the original RoBERTa model by adding the layout term. We also experiment with completely removing the sequential embedding term. Secondly, we apply relative attention bias, used [11, 23, 26] in the context of sequential position. The final architecture is depicted in Fig. 1.

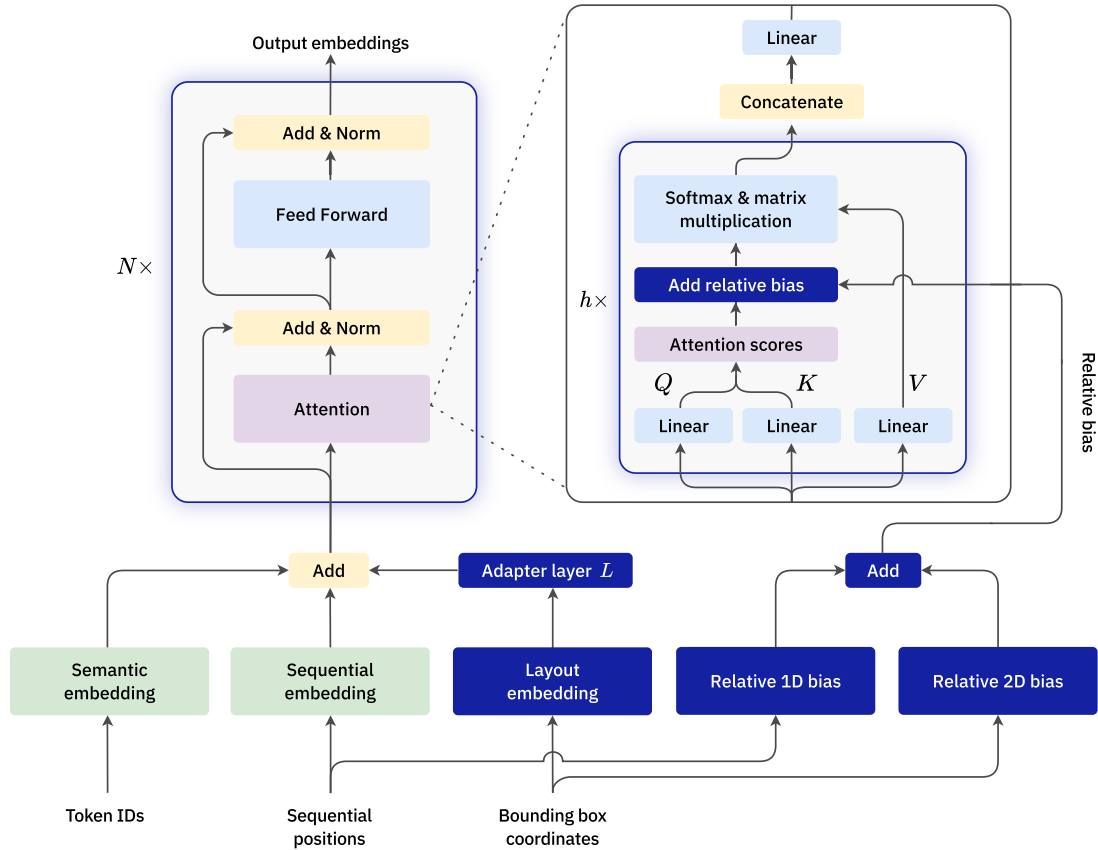


Fig. 1. LAMBERT model architecture. Differences with the plain RoBERTa model are indicated by white text on dark blue background. $N = 12$ is the number of transformer encoder layers, and $h = 12$ is the number of attention heads in each encoder layer. Q , K , and V are, respectively, the queries, keys and values obtained by projecting the self-attention inputs. (Color figure online)

2.1 Background

The basic Transformer encoder, used in, for instance, BERT [6] and RoBERTa [18], is a sequence-to-sequence model transforming a sequence of input embeddings $x_i \in \mathbb{R}^n$ into a sequence of output embeddings $y_i \in \mathbb{R}^m$ of the same length, for the input/output dimensions n and m . One of the main distinctive features of this architecture is that it discards the order of its input vectors. This allows parallelization levels unattainable for recurrent neural networks.

In such a setting, the information about the order of tokens is preserved not by the structure of the input. Instead, it is explicitly passed to the model, by defining the input embeddings as

$$x_i = s_i + p_i, \quad (1)$$

where $s_i \in \mathbb{R}^n$ is the semantic embedding of the token at position i , taken from a trainable embedding layer, while $p_i \in \mathbb{R}^n$ is a *positional embedding*, depending only on i . In order to avoid confusion, we will, henceforth, use the term *sequential embeddings* instead of *positional embeddings*, as the *positional* might be understood as relating to the 2-dimensional position on the page, which we will deal with separately.

Since in RoBERTa, on which we base our approach, the embeddings p_i are trainable, the number of pretrained embeddings (in this case 512) defines a limit on the length of the input sequence. In general, there are many ways to circumvent this limit, such as using predefined [28] or relative [4] sequential embeddings.

2.2 Modification of Input Embeddings

We replace the input embeddings defined in (1) with

$$x_i = s_i + p_i + L(\ell_i). \quad (2)$$

Here, $\ell_i \in \mathbb{R}^k$ stands for *layout embeddings*, which are described in detail in the next subsection. They carry the information about the position of the i -th token on the page.

The dimension k of the layout embeddings is allowed to differ from the input embedding dimension n , and this difference is dealt with by a trainable linear layer $L: \mathbb{R}^k \rightarrow \mathbb{R}^n$. However, our main motivation to introduce the adapter layer L was to gently increase the strength of the signal of layout embeddings during training. In this way, we initially avoided presenting the model with inputs that it was not prepared to deal with. Moreover, in theory, in the case of non-trainable layout embeddings, the adapter layer may be able to learn to project ℓ_i onto a subspace of the embedding space that reduces interference with the other terms in (2). For instance, it is possible for the image of the adapter layer to learn to be approximately orthogonal to the sum of the remaining terms. This would minimize any information loss caused by adding multiple vectors. While this was our theoretical motivation, and it would be interesting to investigate in detail how much of it actually holds, such detailed considerations of a single model component exceed the scope of this paper. We included the impact of using the adapter layer in the ablation study.

We initialize the weight matrix of L according to a normal distribution $\mathcal{N}(0, \sigma^2)$, with the standard deviation σ being a hyperparameter. We have to choose σ carefully, so that in the initial phase of training, the $L(\ell_i)$ term does not interfere overly with the already learned representations. We experimentally determined the value $\sigma = 0.02$ to be near-optimal².

² We tested the values 0.5, 0.1, 0.02, 0.004, and 0.0008.

2.3 Layout Embeddings

In our setting, a document is represented by a sequence of tokens t_i and their bounding boxes b_i . To each element of this sequence, we assign its layout embedding ℓ_i , carrying the information about the position of the token with respect to the whole document. This could be performed in various ways. What they all have in common is that the embeddings ℓ_i depend only on the bounding boxes b_i and not on the tokens t_i .

We base our layout embeddings on the method originally used in [7], and then in [28] to define the sequential embeddings. We first normalize the bounding boxes by translating them so that the upper left corner is at $(0, 0)$, and dividing their dimensions by the page height. This causes the page bounding box to become $(0, 0, w, 1)$, where w is the normalized width.

The layout embedding of a token will be defined as the concatenation of four embeddings of the individual coordinates of its bounding box. For an integer d and a vector of scaling factors $\theta \in \mathbb{R}^d$, we define the corresponding embedding of a single coordinate t as

$$\text{emb}_\theta(t) = (\sin(t\theta); \cos(t\theta)) \in \mathbb{R}^{2d}, \quad (3)$$

where the sin and cos are performed element-wise, yielding two vectors in \mathbb{R}^d . The resulting concatenation of single bounding box coordinate embeddings is then a vector in \mathbb{R}^{8d} .

In [28, Section 3.5], and subsequently in other Transformer-based models with precomputed sequential embeddings, the sequential embeddings were defined by emb_θ with θ being a geometric progression interpolating between 1 and 10^{-4} . Unlike the sequential position, which is a potentially large integer, bounding box coordinates are normalized to the interval $[0, 1]$. Hence, for our layout embeddings we use larger scaling factors (θ_r), namely a geometric sequence of length $n/8$ interpolating between 1 and 500, where n is the dimension of the input embeddings.

2.4 Relative Bias

Let us recall that in a typical Transformer encoder, a single attention head transforms its input vectors into three sequences: queries $q_i \in \mathbb{R}^d$, keys $k_i \in \mathbb{R}^d$, and values $v_i \in \mathbb{R}^d$. The raw attention scores are then computed as $\alpha_{ij} = d^{-1/2} q_i^T k_j$. Afterwards, they are normalized using softmax, and used as weights in linear combinations of value vectors.

The point of relative bias is to modify the computation of the raw attention scores by introducing a bias term: $\alpha'_{ij} = \alpha_{ij} + \beta_{ij}$. In the sequential setting, $\beta_{ij} = W(i-j)$ is a trainable weight, depending on the relative sequential position of tokens i and j . This form of attention bias was introduced in [23], and we will refer to it as *sequential attention bias*.

We introduce a simple and natural extension of this mechanism to the two-dimensional context. In our case, the bias β_{ij} depends on the relative positions

of the tokens. More precisely, let $C \gg 1$ be an integer resolution factor (the number of cells in a grid used to discretize the normalized coordinates). If $b_i = (x_1, y_1, x_2, y_2)$ is the normalized bounding box of the i -th token, we first reduce it to a 2-dimensional position $(\xi_i, \eta_i) = (Cx_1, C(y_1 + y_2)/2)$, and then define

$$\beta_{ij} = H(\lfloor \xi_i - \xi_j \rfloor) + V(\lfloor \eta_i - \eta_j \rfloor), \quad (4)$$

where $H(\ell)$ and $V(\ell)$ are trainable weights defined for every integer $\ell \in [-C, C]$. A good value for C should allow for a distinction between consecutive lines and tokens, without unnecessarily affecting performance. For a typical document $C = 100$ is enough, and we fix this in our experiments.

This form of attention bias will be referred to as *2D attention bias*. We suspect that it should help in analyzing, say, tables by allowing the learning of relationships between cells.

3 Experiments

All experiments were performed on 8 NVIDIA Tesla V100 32 GB GPUs. As our pretrained base model we used RoBERTa in its smaller, base variant (125M parameters, 12 layers, 12 attention heads, hidden dimension 768). This was also employed as the baseline, after additional training on the same dataset we used for LAMBERT. The implementation and pretrained weights from the `transformers` library [30] were used.

In the LAMBERT model, we used the layout embeddings of dimension $k = 128$, and initialized the adapter layer L with standard deviation $\sigma = 0.02$, as noted in Sect. 2. For comparison, in our experiments, we also included the published version of the LayoutLM model [32], which is of a similar size.

The models were trained on a masked language modeling objective extended with layout information (with the same settings as the original RoBERTa [18]); and subsequently, on downstream information extraction tasks. In the remainder of the paper, these two stages will be referred to as, respectively, *training* and *fine-tuning*.

Training was performed on a collection of PDFs extracted from *Common Crawl* made up of a variety of documents (we randomly selected up to 10 documents from any single domain). The documents were processed with an OCR system, `Tesseract 4.1.1-rc1-7-gb36c`, to obtain token bounding boxes. The final model was trained on the subset of the corpus consisting of business documents with non-trivial layout, filtered by an SVM binary classifier, totaling to approximately 315k documents (3.12M pages). The SVM model was trained on 700 manually annotated PDF files to distinguish between business (e.g. invoices, forms) and non-business documents (e.g. poems, scientific texts).

In the training phase, we used the Adam optimizer with the weight decay fix from [30]. We employed a learning rate scheduling method similar to the one used in [6], increasing the learning rate linearly from 0 to $1e-4$ for the warm-up period of 10% of the training time and then decreasing it linearly to 0. The final model was trained with batch size of 128 sequences (amounting to 64K tokens)

for approximately 1000k steps (corresponding to training on 3M pages for 25 epochs). This took about 5 days to complete a single experiment.

After training our models, we fine-tuned and evaluated them independently on multiple downstream end-to-end information extraction tasks. Each evaluation dataset was split into training, validation and test subsets. The models were extended with a simple classification head on top, consisting of a single linear layer, and fine-tuned on the task of classifying entity types of tokens. We employed early stopping based on the F_1 -score achieved on the validation part of the dataset. We used the Adam optimizer again, but this time without the learning rate warm-up, as it turned out to have no impact on the results.

The extended model operates as a tagger on the token level, allowing for the classification of separate tokens, while the datasets contain only the values of properties that we are supposed to extract from the documents. Therefore, the further processing of output is required. To this end, we use the pipeline described in [27].

Every contiguous sequence of tokens tagged as a given entity type is treated as a recognized entity and assigned a score equal to the geometric mean of the scores of its constituent tokens. Then, every recognized entity undergoes a normalization procedure specific to its general data type (e.g. date, monetary amount, address, etc.). This is performed using regular expressions: for instance, the date `July, 15th 2013` is converted to `2013-07-15`. Afterwards, duplicates are aggregated by summing their scores, leading to a preference for entities detected multiple times. Finally, the highest-scoring normalized entity is selected as the output of the information extraction system. The predictions obtained this way are compared with target values provided in the dataset using F_1 -score as the evaluation metric. See [27] for more details.

4 Results

We evaluated our models on four public datasets containing visually rich documents. The Kleister NDA and Kleister Charity datasets are part of a larger Kleister dataset, recently made public [27] (many examples of documents, and detailed descriptions of extraction tasks can be found therein). The NDA set consists of legal agreements, whose layout variety is limited. It should probably be treated as a plain-text dataset. The Charity dataset on the other hand contains reports of UK charity organizations, which include various tables, diagrams and other graphic elements, interspersed with text passages. All Kleister datasets come with predefined train/dev/test splits, with 254/83/203 documents for NDA and 1729/440/609 for Charity.

The SROIE [12] and CORD [21] datasets are composed of scanned and OCR'd receipts. Documents in SROIE are annotated with four target entities to be extracted, while in CORD there are 30 different entities. We use the public 1000 samples from the CORD dataset with the train/dev/test split proposed by the authors of the dataset (respectively, 800/100/100). As for SROIE, it consists of a public training part, and test part with unknown annotations. For the purpose of ablation studies, we further subdivided the public part of SROIE into

Table 1. Comparison of F_1 -scores for the considered models. Best results in each column are indicated in bold. In parentheses, the length of training of our models, expressed in non-unique pages, is presented for comparison. For RoBERTa, the first row corresponds to the original pretrained model without any further training, while in the second row the model was trained on our dataset. ^aresult obtained from relevant publication; ^bresult of a single model, obtained from the SROIE leaderboard [13]

Model	Params	Our experiments				External results	
		NDA	Charity	SROIE*	CORD	SROIE	CORD
RoBERTa [18]	125M	77.91	76.36	94.05	91.57	92.39 ^b	–
RoBERTa (16M)	125M	78.50	77.88	94.28	91.98	93.03 ^b	–
LayoutLM [32]	113M	77.50	77.20	94.00	93.82	94.38 ^a	94.72 ^a
	343M	79.14	77.13	96.48	93.62	97.09 ^b	94.93 ^a
LayoutLMv2 [31]	200M	–	–	–	–	96.25 ^a	94.95 ^a
	426M	–	–	–	–	97.81 ^b	96.01^a
LAMBERT (16M)	125M	80.31	79.94	96.24	93.75	–	–
LAMBERT (75M)	125M	80.42	81.34	96.93	94.41	98.17^b	–

training and test subsets (546/80 documents; due to the lack of a validation set in this split, we fine-tuned for 15 epochs instead of employing early stopping). We refer to this split as SROIE*, while the name SROIE is reserved for the original SROIE dataset, where the final evaluation on the test set is performed through the leaderboard [13].

In Table 1, we present the evaluation results achieved on downstream tasks by the trained models. With the exception of the Kleister Charity dataset, where only 5 runs were made, each of the remaining experiments were repeated 20 times, and the mean result was reported. We compare LAMBERT with baseline RoBERTa (trained on our dataset) and the original RoBERTa [18] (without additional training); LayoutLM [32]; and LayoutLMv2 [31]. The LayoutLM model published by its authors was plugged into the same pipeline that we used for LAMBERT and RoBERTa. In the first four columns we present averaged results of our experiments, and for CORD and SROIE we additionally provide the results reported by the authors of LayoutLM, and presented on the leaderboard [13].

Since the LayoutLMv2 model was not publicly available at the time of preparing this article, we could not perform experiments ourselves. As a result some of the results are missing. For CORD, we present the scores given in [31], where the authors did not mention, though, whether they averaged over multiple runs, or used just a single model. A similar situation occurs for LayoutLM; we presented the average results of 20 runs (best run of LAMBERT attained the score of 95.12), which are lower than the scores presented in [31]. The difference could be attributed to using a different end-to-end evaluation pipeline, or averaging (if the results in [31, 32] come from a single run).

For the full SROIE dataset, most of the results were retrieved from the public leaderboard [13], and therefore they come from a single model. For the base variants of LayoutLM and LayoutLMv2, the results were unavailable, and we present the scores from the corresponding papers.

In our experiments, the base variant of LAMBERT achieved top scores for all datasets. However, in the case of CORD, the result reported in [31] for the large variant of LayoutLMv2 is superior. If we consider the best scores of LAMBERT (95.12) instead of the average, and the scores of LayoutLM reported in [32], LAMBERT slightly outperforms LayoutLM, while still being inferior to LayoutLMv2. Due to the lack of details on the results of LayoutLM, it is unknown which of these comparisons is valid.

For Kleister datasets, the base variant (and in the case of Charity, also the large variant) of LayoutLM did not outperform the baseline RoBERTa. We suspect that this might be the result of LayoutLM being better attuned to the evaluation pipeline used by its authors, and the fact that it was based on an uncased language model. In the Kleister dataset, meanwhile, performance for entities such as names may depend on casing.

5 Hyperparameters and Ablation Studies

In order to investigate the impact of our modifications to RoBERTa, we performed an extensive study of hyperparameters and the various components of the final model. We investigated the dimension of layout embeddings, the impact of the adapter layer L , the size of training dataset, and finally we performed a detailed ablation study of the embeddings and attention biases we had used to augment the baseline model.

In the studies, every model was fine-tuned and evaluated 20 times on each dataset, except for Kleister Charity dataset, on which we fine-tuned every model 5 times: evaluations took much longer on Kleister Charity. For each model and dataset combination, the mean score was reported, together with the two-sided 95% confidence interval, computed using the corresponding t -value. We considered differences to be significant when the corresponding intervals were disjoint. All the results are presented in Table 2, which is divided into sections corresponding to different studies. The F_1 -scores are reported as *increases* with respect to the reported mean baseline score, to improve readability.

5.1 Baseline

As a baseline for the studies we use the publicly available pretrained base variant of the RoBERTa model with 12 layers, 12 attention heads, and hidden dimension 768. We additionally trained this model on our training set, and fine-tuned it on the evaluation datasets in a manner analogous to LAMBERT.

Table 2. Improvements of F_1 -score over the baseline for various variants of LAMBERT model. The first row (with grayed background) contains the F_1 -scores of the baseline RoBERTa model. The other grayed row corresponds to full LAMBERT. Statistically insignificant improvements over the baseline are grayed. In each of three studies, the best result together with all results insignificantly smaller are in bold. ^afiltered datasets; ^bmodel with a disabled adapter layer

Train epochs and pages	Embeddings dimension	Inputs				Datasets				
		Sequential	Seq. bias	Layout	2D bias	NDA	Charity	SROIE*	CORD	
8×2M	128	•				78.50±1.16	77.88±0.48	94.28±0.42	91.98±0.62	
		•	•			2.42 ±0.61	0.52±0.64	0.79±0.17	0.03±0.57	
				•		1.25±0.59	2.62 ±0.80	1.86 ±0.15	0.89±0.83	
					•	-0.49±0.62	2.02±0.48	0.53±0.28	-0.17±0.62	
					•	0.88±0.50	3.00 ±0.37	1.94 ±0.16	0.68±0.62	
				•	•	1.74±0.67	0.06±0.93	1.94 ±0.18	1.42 ±0.53	
				•		1.73±0.60	2.02±0.53	2.09 ±0.22	1.93 ±0.71	
				•	•	0.54±0.85	1.84±0.42	2.08 ±0.38	2.15 ±0.65	
				•	•	1.66±0.76	0.32±1.35	1.75 ±0.35	1.06±0.54	
				•	•	0.85±0.91	1.84±0.27	2.01 ±0.24	1.95 ±0.46	
				•	•	1.81 ±0.60	2.06 ±0.69	1.96 ±0.16	1.77 ±0.46	
		8×2M	128	•	•		1.74±0.67	0.06±0.93	1.94 ±0.18	1.42 ±0.53
			384	•	•		0.90±0.54	0.70±0.40	1.86 ±0.22	1.51 ±0.60
			768	•	•		0.71±1.04	0.50±0.85	2.18 ±0.25	1.54 ±0.51
768 ^b	•		•		0.77±0.58	2.30 ±0.20	0.37±0.15	1.58 ±0.52		
8×2M		•	•	•	•	1.81 ±0.60	2.06±0.26	1.96±0.18	1.77±0.46	
8×2M ^a	128	•	•	•	•	1.86 ±0.66	1.92±0.19	2.60 ±0.18	1.59±0.61	
25×3M ^a		•	•	•	•	1.92 ±0.50	3.46 ±0.21	2.65 ±0.13	2.43 ±0.19	

5.2 Embeddings and Biases

In this study we disabled various combinations of input embeddings and attention biases. The models were trained on 2M pages for 8 epochs, with 128-dimensional layout embeddings (if enabled). The resulting models were divided into three groups. The first one contains sequential-only combinations which do not employ the layout information at all, including the baseline. The second group consists of models using only the bounding box coordinates, with no access to sequential token positions. Finally, the models in the third group use both sequential and layout inputs. In this group we did not disable the sequential embeddings. It includes the full LAMBERT model, with all embeddings and attention biases enabled.

Generally, we observe that none of the modifications has led to a significant performance deterioration. Among the models considered, the only one which

reported a significant improvement for all four datasets—and at the same time, the best improvement—was the full LAMBERT.

For the Kleister datasets the variance in results was relatively higher than in the case of SROIE* and CORD. This led to wider confidence intervals, and reduced the number of significant outcomes. This is true especially for the Kleister NDA dataset, which is the smallest one. In Kleister NDA, significant improvements were achieved for both sequential-only models, and for full LAMBERT. The differences between these increases were insignificant. It would seem that, for sequential-only models, the sequential attention bias is responsible for the improvement. But after adding the layout inputs, it no longer leads to improvements when unaccompanied by other modifications. Still, achieving better results on sequential-only inputs may be related to the plain text nature of the Kleister NDA dataset.

While other models did not report significant improvement over the baseline, there are still some differences between them to be observed. The model using only 2D attention bias is significantly inferior to most of the others. This seems to agree with the intuition that relative 2D positions are the least suitable way to pass positional information about plain text.

In the case of the Kleister Charity dataset, significant improvements were achieved by all layout-only models, and all models using the 2D attention bias. Best improvement was attained by full LAMBERT, and two layout-only models using the layout embeddings; the 2D attention bias used alone improved the results significantly, but did not reach the top score. The confidence intervals are too wide to offer further conclusions, and many more experiments will be needed to increase the significance of the results.

For the SROIE* dataset, except for two models augmented only with a single attention bias, all improvements proved significant. Moreover, the differences between all the models using layout inputs are insignificant. We may conclude that passing bounding box coordinates in any way, except through 2D attention bias, significantly improves the results. As to the lack of significant improvements for 2D attention bias, we hypothesize that this is due to its relative nature. In all other models the absolute position of tokens is somehow known, either through the layout embeddings, or the sequential position. When a human reads a receipt, the absolute position is one of the main features used to locate the typical positions of entities.

For CORD, which is the more complex of the two receipt datasets, significant improvements were observed only for combined sequential and layout models. In this group, the model using both sequential and layout embeddings, augmented with sequential attention bias, did not yield a significant improvement. There were no significant differences among the remaining models in the group. Contrary to the case of SROIE*, none of the layout-only models achieved significant improvement.

5.3 Layout Embedding Dimension

In this study we evaluated four models, using both sequential and layout embeddings, varying the dimension of the latter. We considered 128-, 384-, and 768-dimensional embeddings. Since this is the same as for the input embeddings of RoBERTa_{BASE}, it was possible to remove the adapter layer L , and treat this as another variant, in Table 2 denoted as 768^b.

In Kleister NDA, there were no significant differences between any of the evaluated models, and no improvements over the baseline. On the other hand, in Kleister Charity, disabling the adapter layer and using the 768-dimensional layout embeddings led to significantly better performance. These results remain consistent with earlier observations that in Kleister NDA the best results were achieved by sequential-only models, while in the case of Kleister Charity, by layout-only models. It seems that in the case of NDA the performance is influenced mostly by the sequential features, while in the case of Charity, removing the adapter layer increases the strength of the signal of the layout embeddings, carrying the layout features which are the main factor affecting performance.

In SROIE* and CORD all results were comparable, with one exception, namely in SROIE*, the model with the disabled adapter layer did not, unlike the remaining models, perform significantly better than the baseline.

5.4 Training Dataset Size

In this study, following the observations from [9], we considered models trained on 3 different datasets. The first model was trained for 8 epochs on 2M unfiltered (see Sect. 3 for more details of the filtering procedure) pages. In the second model, we used the same training time and dataset size, but this time only filtered pages were used. Finally, the third model was trained for 25 epochs on 3M filtered pages.

It is not surprising that increasing the training time and dataset size, leads to an improvement in results, at least up to a certain point. In the case of Kleister NDA dataset, there were no significant differences in the results. For Kleister Charity, the best result was achieved for the largest training dataset, with 75M filtered pages. This result was also significantly better than the outcomes for the smaller dataset. In the case of SROIE* the two models trained on datasets with filtered documents achieved a significantly higher score than the one trained on unfiltered documents. There was, in fact, no significant difference between these two models. This supports the hypothesis that, in this case, filtering could be the more important factor. Finally, for CORD the situation is similar to Kleister Charity.

6 Conclusions and Further Research

We introduced LAMBERT, a layout-aware language model, producing contextualized token embeddings for tokens contained in formatted documents. The

model can be trained in an unsupervised manner. For the end user, the only difference with classical language models is the use of bounding box coordinates as additional input. No images are needed, which makes this solution particularly simple, and easy to include in pipelines that are already based on OCR-ed documents.

The LAMBERT model outperforms the baseline RoBERTa on information extraction from visually rich documents, without sacrificing performance on documents with a flat layout. This can be clearly seen in the results for the Kleister NDA dataset. Its base variant with around 125M parameters is also able to compete with the large variants of LayoutLM (343M parameters) and LayoutLMv2 (426M parameters), with Kleister and SROIE datasets achieving superior results. In particular, LAMBERT_{BASE} achieved first place on the Key Information Extraction from the SROIE dataset leaderboard [13].

The choice of particular LAMBERT components is supported by an ablation study including confidence intervals, and is shown to be statistically significant. Another conclusion from this study is that for visually rich documents the point where no more improvement is attained by increasing the training set has not yet been reached. Thus, LAMBERT’s performance can still be improved upon by simply increasing the unsupervised training set. In the future we plan to experiment with increasing the model size, and training datasets.

Further research is needed to ascertain the impact of the adapter layer L on the model performance, as the results of the ablation study were inconclusive. It would also be interesting to understand whether the mechanism through which it affects the results is consistent with the hypotheses formulated in Sect. 2.

Acknowledgments. The authors would like to acknowledge the support the Applica.ai project has received as being co-financed by the European Regional Development Fund (POIR.01.01.01–00–0605/19–00).

References

1. Amazon: Amazon Textract (2019). <https://aws.amazon.com/textract/>. Accessed 25 Nov 2019
2. Bart, E., Sarkar, P.: Information extraction by finding repeated structure. In: DAS 2010 (2010)
3. Cesarini, F., Francesconi, E., Gori, M., Soda, G.: Analysis and understanding of multi-class invoices. *IJDAR* **6**, 102–114 (2003)
4. Dai, Z., et al.: Transformer-XL: Attentive language models beyond a fixed-length context. In: *ACL* (2019)
5. Denk, T.I., Reisswig, C.: BERTgrid: contextualized Embedding for 2D document representation and understanding. In: *Workshop on Document Intelligence at NeurIPS 2019* (2019)
6. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: pre-training of deep bidirectional transformers for language understanding. In: *NAACL-HLT* (2019)
7. Gehring, J., Auli, M., Grangier, D., Yarats, D., Dauphin, Y.N.: Convolutional sequence to sequence learning. In: *ICML* (2017)

8. Google: Cloud Document Understanding AI (2019). <https://cloud.google.com/document-understanding/docs/>. Accessed 25 Nov 2019
9. Gururangan, S., Marasović, A., Swayamdipta, S., Lo, K., Beltagy, I., Downey, D., Smith, N.A.: Don't stop pretraining: adapt language models to domains and tasks. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, pp. 8342–8360. Association for Computational Linguistics (2020). <https://doi.org/10.18653/v1/2020.acl-main.740>
10. Hamza, H., Belaid, Y., Belaid, A., Chaudhuri, B.: An end-to-end administrative document analysis system. In: 2008 The Eighth IAPR International Workshop on Document Analysis Systems, pp. 175–182 (2008)
11. Huang, Y., et al.: Gpipe: Efficient training of giant neural networks using pipeline parallelism. In: NeurIPS (2019)
12. ICDAR: Competition on Scanned Receipts OCR and Information Extraction (2019). <https://rrc.cvc.uab.es/?ch=13>. Accessed 21 Feb 2021
13. ICDAR: Leaderboard of the Information Extraction Task, Robust Reading Competition (2020). <https://rrc.cvc.uab.es/?ch=13&com=evaluation&task=3>. Accessed 7 Apr 2020
14. Ishitani, Y.: Model-based information extraction method tolerant of ocr errors for document images. *Int. J. Comput. Process. Orient. Lang.* **15**, 165–186 (2002)
15. Katti, A.R., et al.: Chargrid: towards understanding 2D documents. In: EMNLP (2018)
16. Lewis, D., Agam, G., Argamon, S., Frieder, O., Grossman, D., Heard, J.: Building a test collection for complex document information processing. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (2006)
17. Liu, X., Gao, F., Zhang, Q., Zhao, H.: Graph convolution for multimodal information extraction from visually rich documents. In: NAACL-HLT (2019)
18. Liu, Y., et al.: RoBERTa: A Robustly Optimized BERT Pretraining Approach. *ArXiv arXiv:1907.11692* (2019)
19. Medvet, E., Bartoli, A., Davanzo, G.: A probabilistic approach to printed document understanding. *IJDAR* **14**, 335–347 (2011)
20. Microsoft: Cognitive Services (2019). <https://azure.microsoft.com/en-us/services/cognitive-services/>. Accessed 25 Nov 2019
21. Park, S., et al.: CORD: A Consolidated Receipt Dataset for Post-OCR Parsing. In: Document Intelligence Workshop at Neural Information Processing Systems (2019)
22. Peanho, C., Stagni, H., Silva, F.: Semantic information extraction from images of complex documents. *Appl. Intell.* **37**, 543–557 (2012)
23. Raffel, C., et al.: Exploring the limits of transfer learning with a unified text-to-text transformer. *J. Mach. Learn. Res.* **21**(140), 1–67 (2020)
24. Rahman, W., et al.: Integrating multimodal information in large pretrained transformers. In: ACL (2020)
25. Rusinol, M., Benkhelfallah, T., Poulain d'Andecy, V.: Field extraction from administrative documents by incremental structural templates. In: ICDAR (2013)
26. Shaw, P., Uszkoreit, J., Vaswani, A.: Self-attention with relative position representations. In: NAACL-HLT (2018)
27. Stanisławek, T., et al.: Kleister: A novel task for information extraction involving long documents with complex layout (2021) . *ArXiv arXiv:2105.05796* Accepted to ICDAR 2021
28. Vaswani, A., et al.: Attention is all you need. In: Advances in Neural Information Processing Systems 30 (2017)

29. Wang, A., Singh, A., Michael, J., Hill, F., Levy, O., Bowman, S.R.: GLUE: a multi-task benchmark and analysis platform for natural language understanding. In: Proceedings of ICLR (2019). <https://gluebenchmark.com/>. Accessed 26 Nov 2019
30. Wolf, T., et al.: Transformers: state-of-the-art natural language processing. In: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, pp. 38–45. Association for Computational Linguistics, Online (October 2020). <https://www.aclweb.org/anthology/2020.emnlp-demos.6>
31. Xu, Y., et al.: LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. arXiv [arXiv:2012.14740](https://arxiv.org/abs/2012.14740) (2020)
32. Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., Zhou, M.: LayoutLM: pre-training of text and layout for document image understanding. In: Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining, pp. 1192–1200 (2020)
33. Yu, W., Lu, N., Qi, X., Gong, P., Xiao, R.: PICK: Processing key information extraction from documents using improved graph learning-convolutional networks. In: 2020 25th International Conference on Pattern Recognition (ICPR), pp. 4363–4370 (2021). <https://doi.org/10.1109/ICPR48806.2021.9412927>
34. Zhang, P., et al.: TRIE: end-to-end text reading and information extraction for document understanding. In: Proceedings of the 28th ACM International Conference on Multimedia (2020)

STable Table Generation Framework for Encoder-Decoder Models

Michał Pietruszka^{*1, 2}, Michał Turski^{*1, 3}, Łukasz Borchmann^{*1}, Tomasz Dwojak¹,
Gabriela Pałka^{1, 3}, Karolina Szyndler¹, Dawid Jurkiewicz^{1, 3}, and Łukasz Garncarek¹

¹Snowflake

²Jagiellonian University

³Adam Mickiewicz University

Abstract

Since the output structure of database-like tables can cover a wide range of NLP tasks, we propose a framework for text-to-table neural models applicable to, e.g., extraction of line items, joint entity and relation extraction, or knowledge base population. The permutation-based decoder of our proposal is a generalized sequential method that comprehends information from all cells in the table. The training maximizes the expected log-likelihood for a table’s content across all random permutations of the factorization order. During the content inference, we exploit the model’s ability to generate cells in any order by searching over possible orderings to maximize the model’s confidence and avoid substantial error accumulation, which other sequential models are prone to. Experiments demonstrate a high practical value of the framework, which establishes state-of-the-art results on several challenging datasets, outperforming previous solutions by up to 15%.

1 Introduction

It has been previously shown that encoder-decoder models are capable of unifying a variety of problems involving natural language. In this setting, unification is achieved by casting different tasks as Question Answering with a plain-text answer, i.e., assuming the text-to-text (Kumar et al., 2016; Raffel et al., 2020; McCann et al., 2018; Khashabi et al., 2020) or document-to-text scenario (Powalski et al., 2021; Kim et al., 2022). We argue that the restriction of output type to raw text is suboptimal for the plethora of NLP problems and propose a decoder architecture able to infer *aggregate* data types such as a list of ordered tuples or a database-like table (see Figure 1).

Though the encoder-decoder architecture was formerly used to infer lists (Powalski et al., 2021),

named tuples (Dwojak et al., 2020), or even more complex structures (Townsend et al., 2021), it was often achieved in an autoregressive manner, without any architectural changes. A model intended for the generation of *unstructured* text in natural language was used to infer an output with formal *structure*. In contrast, we exploit regularities and relationships within the output data and employ a grammar-constrained decoding process (Section 2.5).

Specifically, we focus on the text-to-table inference with applications to problems such as extraction of line items, key information extraction of multiple properties, joint entity and relation extraction, or knowledge base population. Tables as we understand them are equivalent to database tables and defined as a set of values structured in horizontal rows and vertical columns identifiable by name.

From receipts and invoices, through paycheck stubs and insurance loss run reports, to scientific articles, real-world documents contain explicitly or implicitly tabular data to be extracted. These are not necessarily represented as a table *per se* within the input document, e.g., the currency name on the invoice or policy number on the loss run can be mentioned once and be related to all the line items within. In other cases, the evidence one intends to comprehend and represent as a table may be available in free-text only, as can be found in problems of joint entity and relation extraction (see Figure 1-2). Finally, the data may require some postprocessing, such as the normalization of dates, before returning them to the end-user.

1.1 Limitation of Current Approaches

Admittedly, models based on the transformer encoder-decoder or decoder achieve remarkable results in generating complex, formalized outputs, such as computer programs or JSON files (Chen et al., 2021; Townsend et al., 2021). Nevertheless, we hypothesize that changes leading to the *explicit*

* equal contribution
firstname.lastname@snowflake.com

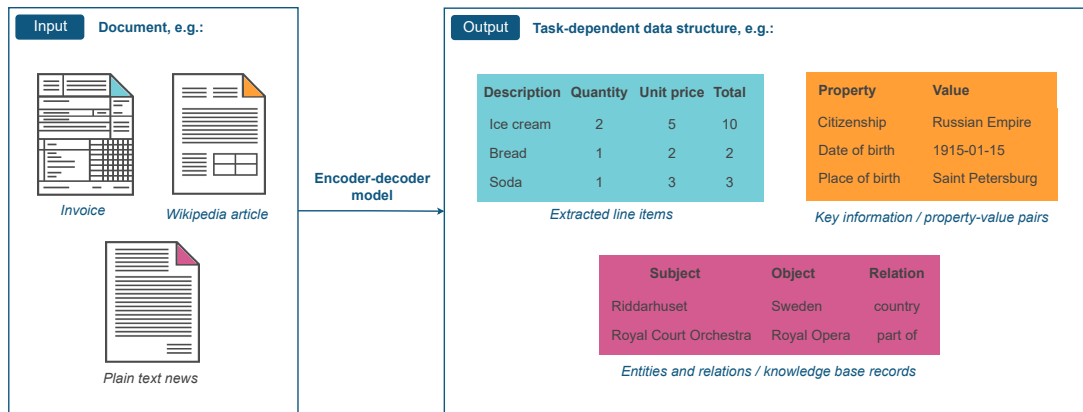


Figure 1: Reinterpreting diverse tasks under a unified paradigm: all these tasks essentially require generating a table based on a given context. While they were not previously seen in this light, we reinterpret them as text-to-table tasks, bringing them together under a single paradigm and directly model the table in the output. This unification has led to significant improvements in each task.

modeling of structured data can outperform the said *implicit* decoding that models long-range syntax dependencies sequentially and does not guarantee the formal validity of produced outputs.

While generating in a particular predefined order (e.g., left-to-right, row-by-row), such approaches have a few drawbacks. Firstly, error propagation that causal models may show after skipping some cells or answering them incorrectly. This flaw may start a chain reaction and directly influence the subsequent cells’ generation, causing error propagation and a rapid decline in table quality. Strikingly, an error propagation issue is known in Neural Machine Translation when the right part of the generated sentence used to be worse than the left one (Wu et al., 2018). Therefore, previous approaches to table generation employed preventive measures to keep the table layout under control (Wang et al., 2019) and limit the negative effect of error propagation. Secondly, the answers are forced; the model that cannot give a proper answer consistently has lower confidence and dispersed probability over multiple possibilities. Therefore, we use logit-based confidence to guide the generation process, emergently achieving the property of abstaining from generating answers when the model does not indicate high confidence. Thirdly, the formatting of the table plays a role, and the order of columns may be treated as a hyperparameter in the previous approaches (Wang et al., 2019; Dwojak et al., 2020). For example, performing generation in a predefined and not optimized order may lead to the case when the model is asked about, e.g., date of birth of the person that still needs to be

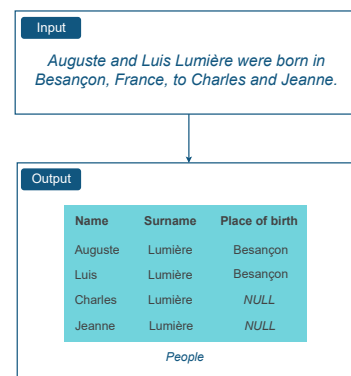


Figure 2: Example of text-to-table generation given plain text input. Concurrent extraction and grouping of the detected entities simplifies the process and may mitigate error accumulation.

specified. Therefore, we want the model to learn the optimal order of the generation as part of the task itself without any implicit human guidance.

Significantly, the advantage the encoder-decoder framework has is that it can cover problems mentioned above in one end-to-end trainable process, thus simplifying the pipeline and reducing the accumulation of errors along the way. At the same time, since extracted data is already in the form the end user requires, one is able to use it directly for downstream application without further processing steps.

1.2 Contribution and Related Works

The specific contribution of this work includes (1) equipping transformer models with permutation-based decoder training to allow comprehending complex, role-dependent relationships in a series of similar objects we represent as a table,

(2) a sequential, grammar-constrained decoding mechanism which generates table content cell-by-cell, in a dynamic, data-dependent order, and (3) introduction of tabular attention bias to the decoder. The novelty of our approach can be better understood in the context of related works.

Decoding of data structures. A few authors attempted the problem of table generation in the encoder-decoder framework. [Zhong et al. \(2020\)](#) proposed a table recognition model consuming input images and decoupled the problem into unconstrained table and cell content generation. In comparison, (1) we use a single constrained decoder comprehending both table structure and its content; (2) we tackle problems of text-to-table inference where the presence of a table at the model input is optional. Recently, [Wu et al. \(2022\)](#) introduced a model relying on constrained decoding of table and tabular embeddings similar to ours. We share their motivation and idea but differ as (1) our method is not restricted to a predefined, row-by-row decoding order and uses a permutation-based training procedure aligned with the use of optimal, model-guided cell permutation during inference; (2) we assume the explicit prediction of the number of rows upfront (before the table decoding starts), instead of allowing the model to stop the generation process after any completed row. The advantage of this approach is discussed in Section 2 and proven by a series of experiments reported in Section 3.

The encoder-decoder model was previously used *as is*, to infer lists and tuples separated with special characters ([Powalski et al., 2021](#); [Dwojak et al., 2020](#)). Similarly, [Townsend et al. \(2021\)](#) experimented with the generation of more complex data types represented as XML, JSON, or Python’s string representation. In contrast to previous approaches, we do not rely on *implicit* modeling of the formal structure of the output but opt for *explicit* structure generation.

Finally, a text-to-structure approach was recently taken by [Lu et al. \(2021\)](#) for event extraction. The authors used trie-based constrained decoding with event schema injected as the decoder prompt. It resembles our approach to constrained table generation, though they rely on only one proper decoding order resulting from the assumed tree linearization strategy. Moreover, the authors found it challenging to train the structure generation model directly and thus trained it on simple event substructures first. In contrast, we can directly train

the structure decoder, and our permutation-based method allows one to generate the structure *flexibly*, in an arbitrary order dynamically guided by the decoding algorithm.

Flexible generation. Even though permutation-based training, which allows for output generation in any order, is of minor usability in the task of LM, it was validated by [Stern et al. \(2019\)](#) for machine translation and by [Song et al. \(2021\)](#) for summarization. Accordingly, [Stern et al. \(2019\)](#) proposed to equip a transformer with the insertion operation, realized by interpreting an additional number generated with the token as the position in the output sequence to which the insertion should be performed. This framework allows for the flexibility of the decoding process, understood as the possibility of stubbing the output sequence with tokens that the model recognizes with high confidence first and then gradually adding more details in the later iterations. In contrast, since the whole output sequence is passed through the decoder anyway, our one cell-decoding step is implemented by sampling all cells at once and then choosing the best-scored ones to be inserted at its location while disregarding others. In the ablation studies we evaluate how the number of cells inserted at once influence the decoding speed and quality, as higher values indicate more cells generated in parallel.

Permutation-based language modeling. The effectiveness of the permutation-based language modeling objective was demonstrated by [Yang et al. \(2019\)](#) who conditioned the BERT-like model to work with the AR objective. However, while the nature of the LM task allowed them to perturb the factorization order of the input sequence arbitrarily, our table-decoding problem requires additional constraints to account for the fact that each cell may consist of several tokens. Thus, the factorization order of blocks of tokens (representing cells) is permuted, while causal order is assumed within the cell. For permutation-invariance and table-awareness on reversed tasks (i.e., table-to-text), we refer the reader to ([Wang et al., 2022](#)).

2 STable — Text-to-Table Framework

Serialized representation of the table permits to treat it as a text sequence, and hence, use text-centric methods to perform an autoregressive generation of the output sequence by employing a vanilla Transformer decoder. However, this approach does

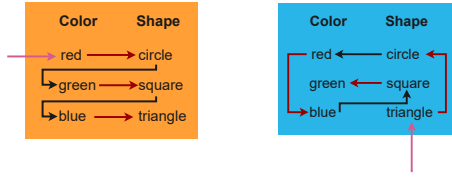


Figure 3: A comparative illustration of the training examples under linearized versus permuted cell ordering. The left panel depicts a typical linearized ordering, following a top-down, left-to-right progression. The right panel presents a permuted ordering example where cells are filled in a non-sequential order.

not exploit the two-dimensional structure of the table as it expands the answer sequentially and utilizes only uni-directional context.

Consequently, two challenging problems arise. Firstly, how to approach the fact that some information in the table may depend on other cells (e.g., name and surname or the same tax rate for similar items on a receipt) while some may not be dependent (prices of different articles on the shopping list). In general, a model possesses flexibility with respect to this dependence-independence assumption when it can leverage dependencies during decoding but is not forced to do so in any specific order. Our idea (presented in Figure 3) is to solve this problem by delaying the generation of the most challenging and complex answers to later stages and conditioning them on the already generated answer.

Moreover, the decoding must remain free of train-inference discrepancies. Generally, the train-inference alignment means that the state of the table at every step while decoding a particular example must also be possible to achieve in the training phase. Formulating the training that allows for flexible cell generation without providing any additional information remains a non-trivial problem. We rise up to the challenge and demonstrate the solution below.

2.1 Decoding Invariant Under Cell Order

Instead of generating the cell values in a top-down, left-to-right manner as previously seen in the literature (e.g., Wu et al., 2022), we perform the pre-training by maximizing the expected log-likelihood of the sequence of cell values over all possible prediction orders. More specifically, suppose that we are given a document containing a table with

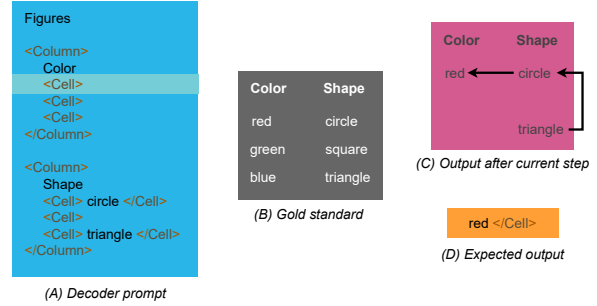


Figure 4: A training example depicting how the answer red is produced based on the partially filled cells containing circle and triangle. (A) The highlighted cell denotes a position where the expected red `</Cell>` should be predicted autoregressively starting from a `<Cell>` token. A successfully decoded cell will lead to the state visible in (C), i.e., the partially decoded gold standard table (B). The generation order of a table is random for each example in the training.

row labels $\mathbf{r} = (r_1, \dots, r_N)$,¹ and column labels $\mathbf{c} = (c_1, \dots, c_M)$, which we will collectively denote $\mathbf{h} = (\mathbf{r}, \mathbf{c})$. A linear ordering of the table cells can be represented with a bijection

$$\sigma: \{1, 2, \dots, C\} \rightarrow \{1, \dots, N\} \times \{1, \dots, M\},$$

where $C = NM$ is the number of cells, so that $\sigma(n) = (i, j)$ are the row and column coordinates of the n -th cell in the ordering. Given such a σ and cell values $\mathbf{v} = (v_{ij})_{i \leq N, j \leq M}$, we factorize the likelihood of \mathbf{v} given \mathbf{h} as

$$p_{\theta}(\mathbf{v}|\mathbf{h}) = \prod_{n=1}^C p_{\theta}(v_{\sigma(n)} | (v_{\sigma(k)})_{k < n}, \mathbf{h}), \quad (1)$$

and using this factorization, we maximize the expected log-likelihood

$$\frac{1}{C!} \sum_{\sigma} \sum_{n=1}^C \log p_{\theta}(v_{\sigma(n)} | (v_{\sigma(k)})_{k < n}, \mathbf{h}) \quad (2)$$

over θ . The likelihoods $p_{\theta}(v_{\sigma(n)} | (v_{\sigma(k)})_{k < n}, \mathbf{h})$ themselves can be factorized according to the standard auto-regressive approach as

$$\begin{aligned} p_{\theta}(v_{\sigma(n)} | (v_{\sigma(k)})_{k < n}, \mathbf{h}) &= \\ &= \prod_{t=1}^{\ell(v_{\sigma(n)})} p_{\theta}(v_{\sigma(n)}^t | (v_{\sigma(n)}^i)_{i < t}, (v_{\sigma(k)})_{k < n}, \mathbf{h}) \end{aligned} \quad (3)$$

¹In practice, usually there are no row labels; however, in the decoder, the special tokens used for distinguishing rows take this role.

where $\ell(v_{\sigma(n)})$ is the length of $v_{\sigma(n)}$ represented as a sequence of tokens $(v_{\sigma(n)}^i)_{i \leq L}$. In practice, the expected log-likelihood is estimated by sampling bijections σ at random.

Training example is presented in Figure 4.

2.2 Tabular Attention Bias

We base our attention computation method on the relative bias idea popularized by the T5 model. Given a text consisting of T tokens, in the vanilla T5 model, raw attention scores α_{ij} for tokens i and j (with $0 \leq i, j < T$) are modified by introducing a bias term: $\alpha'_{ij} = \alpha_{ij} + \beta_{ij}$ where $\beta_{ij} = W(i - j)$ is a trainable weight, depending on the relative sequential position of these tokens (Raffel et al., 2020).

We modify the decoder’s self-attention by extending it with two new bias terms, defined below. The *tabular bias* τ_{ij} encodes the relative position of table cells in which the tokens lie, while the *local sequential bias* λ_{ij} corresponds to the relative sequential position of tokens belonging to the same cell.

$$\tau_{ij} = \begin{cases} R(r_i - r_j) + C(c_i - c_j) & \text{if } r_j > 0 \\ R_0 + C(c_i - c_j) & \text{if } r_j = 0 \end{cases},$$

$$\lambda_{ij} = \begin{cases} L(i - j) & \text{if } (c_i, r_i) = (c_j, r_j) \\ 0 & \text{otherwise} \end{cases} \quad (4)$$

where (c_i, r_i) are cell coordinates as given by its 1-based column and row indices (with 0 reserved for the header row/column), and $R(k)$, $C(k)$, $L(k)$ and R_0 are trainable weights. The special case with $r_j = 0$ corresponds to the situation when the key/value token lies in the column header, in which case we want to use the same bias independent of the row of the query token, due to the different nature of the relation between two cells, and a cell and its column header. After these adjustments, the final attention score takes the form $\alpha'_{ij} = \alpha_{ij} + \beta_{ij} + \tau_{ij} + \lambda_{ij}$, where β_{ij} is the bias term defined earlier.

2.3 Predicting Number of Groups

Although the previous work of Wu et al. (2022) assumed the table is finalized when the appropriate special token explicitly appears in the output, our systematic study shows that the explicit prediction of the number of groups yields better results (see Section 4 for comparison). This explicit prediction is achieved with a linear layer that consumes the first input token’s embedding to perform a predic-

tion on the number of groups. During the training stage, the layer’s output is scored against the known number of groups using MSE loss, while during the inference, it is used as a predictor declaring the number of groups to populate the template with.

2.4 Inference with Model-Guided Cell Order

Since the model was trained assuming a permuted factorization of cell ordering, in expectation, the model learned to understand all possible variants of a partially-filled table and predict values for all empty cells. Because each step in the generation process implicates uncertainty that should be globally minimized, we propose to estimate the optimal table decoding algorithm by greedily finding the cell that minimizes this uncertainty at each step.

The decoding employs an outer loop that progresses cell-by-cell, an inner loop that generates each cell that is yet to render, and a selection heuristics that determine which cell, from all the finalized in the inner loop, should be added to the outer loop. The heuristic we use selects the cell containing the token with highest probability among all predicted (Figure 5). The detailed study of this and alternative selection criteria is presented in Appendix C.

In the inner loop, each cell is decoded until the special token determining the end of cell generation is placed. As the inner loop generates each cell autoregressively and independently from other cells, the process can be treated as generating multiple concurrent threads of an answer and is well parallelizable. In the worst case, it takes as many steps as the number of tokens in the most extended cell.

After being selected by a heuristic, the cell from the inner loop is inserted into the outer loop, and made visible to all other cells, while the cells that were not selected are to be reset and continuously generated in the future steps until they are chosen by a heuristic (see pseudocode in Appendix A).

2.5 Grammar-Constrained Decoding

As a result of the model design, incorrect tables cannot be generated. Part of these rules is explicit (e.g., we overwrite logits, so it is impossible to emit particular tokens such as the end-of-cell when no cell is opened), whereas part of the rules results implicitly from the algorithm (template-filling setting, where the well-formulated table is always ensured).

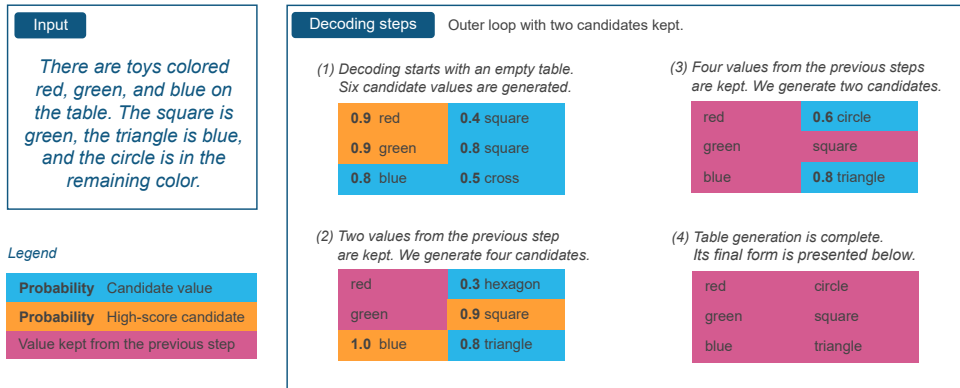


Figure 5: A possible progression of decoding a table given the text on the input. Since the probabilities guide the decoding order, the circle’s color that was not explicitly stated in the text is determined at the last step.

Table 1: Results on public and private datasets assuming task-specific metrics. The results of a sequence-to-sequence baseline that learns and generates tables as text are provided in the *Linearized* column. Mean and STD over three runs. The [†] symbol denotes our TILT training. Underline signifies our model is significantly better than baseline.

Dataset	State-of-the-Art Reference		Linearized		Our Model
PWC★	T5 2D (Borchmann et al., 2021)	26.8	27.8 ± 1.0	30.8 ± 0.5	T5 2D + STable 🐎
CORD	TILT (Powalski et al., 2021)	96.3	92.4 ± 0.7	<u>95.6</u> ± 0.2	TILT [†] + STable 🐎
Rotowire					
Player	Text-to-Table (Wu et al., 2022)	86.8	84.5 ± 0.7	84.5 ± 0.2	T5 + STable 🐎
Team	(BART backbone)	86.3	83.8 ± 0.9	<u>84.7</u> ± 0.2	
DWIE	KB-both (Verlinden et al., 2021)	62.9	60.2 ± 1.5	59.2 ± 1.5	T5 + STable 🐎
Recipe...		71.9	60.1 ± 0.3	75.5 ± 1.6	
Payment...	TILT [†]	77.0	72.0 ± 2.3	79.1 ± 0.9	TILT [†] + STable 🐎
Bank...		61.1	58.7 ± 4.9	69.9 ± 4.8	

3 Experiments

In addition to state-of-the-art reference and our results, we provide scores of the same backbone models (T5, T5 2D, and TILT) while a table linearization strategy follows the assumptions of Wu et al. (2022)’s baselines. Appendix D covers details of training procedure.

Metrics. We rely on the original metrics for all but the DWIE dataset, i.e., GROUP-ANLS for PWC★, F1 for CORD, and non-header exact match cell F1 for Rotowire (other variants proposed by the authors are reported in Table 7 in Appendix D). Use of the original DWIE metric was not possible, as it assumes a step-by-step process. In contrast, we tackle the problem end-to-end, i.e., return (*object*, *relation*, *subject*) tuples without detecting all entity mentions within the document and their locations. To ensure a fair comparison, we use the F1 score calculated on triples; that is, we require the

model to return the exact match of the triple. Such a setup is very demanding for encoder-decoder models as the convention in DWIE is to require *object* and *subject* to be returned in the longest form of appearance in the document.

Pretraining and Adaptation. Due to the switch to permutative training and the addition of the regression head, there is a significant change in the model objective. Consequently, we anticipated the necessity of the model adaptation phase. It consists of the pretraining stage equivalent to the one conducted by authors of the TILT model (Powalski et al., 2021) extended by Natural Questions (Kwiatkowski et al., 2019) and WebTables² datasets. To utilize WebTables we rendered webpages, from which the tables were scraped and taught models to extract table contents from webpages. The said stage is applied to all T5+STable, T5 2D+STable, and TILT+STable models.

²<https://webdatacommons.org/webtables/>

Complex Information Extraction. The problem of information extraction involving aggregated data types, where one may expect improvement within the document-to-table paradigm, is prevalent in business cases. Nevertheless, the availability of public datasets here is limited to PWC★ (Borchmann et al., 2021; Kardas et al., 2020) and CORD (Park et al., 2019).

In the case of PWC★, the goal is to determine model names, metrics, datasets, and performance, given the machine learning paper as an input. CORD assumes the extraction of line items from images of Indonesian receipts, among others. To determine the gain from our STable decoder, the experiments are conducted with state-of-the-art encoder-decoder models proposed for these datasets (T5 2D and TILT), assuming the same training procedure (Borchmann et al. (2021); Powalski et al. (2021); see Appendix D for details).

Additionally, due to the sparsity of public benchmarks of this kind, we decided to provide results on three confidential datasets. They assume, respectively, (1) the extraction of payments’ details from *Payment Stubs*, (2) *Recipe Composition* from documents provided by a multinational snack and beverage corporation, as well as (3) account balances from *Bank Statements*. These are covered in details in Appendix E and addressed by the TILT+STable model with vanilla TILT as a reference.

As summarized in Table 1, we outperformed state-of-the-art information extraction models on several datasets. At the same time, the CORD where we underperform was previously considered solved, e.g., Powalski et al. (2021) point that TILT’s output and the reference differed insignificantly. We used it in the experiment as a safety check to determine whether the model can maintain almost-perfect scores after applying the STable decoder. Consequently, we omit it in the ablation studies.

The rest of the experiments were conducted assuming the vanilla T5 model (Raffel et al., 2020) equipped with the STable decoder of our proposal.

Joint Entity and Relation Extraction. To demonstrate the broad applicability of the model, we consider the problem of a joint entity and relation extraction on the example of the DWIE dataset (Zaporojets et al., 2021). Here, the tuples consisting of entities and one of the sixty-five relation types are to be determined given a plain-text news article. Despite not outperforming a multi-step state-of-the-art model, we achieved

high scores and were the first to prove that the problem can be successfully approached end-to-end using an encoder-decoder framework. Here, the T5+STable’s errors and issues reflect the very demanding assumptions of DWIE, where it is required to return *object* and *subject* in the longest form of appearance in the document.

Reversed Table-to-Text. Finally, following Wu et al. (2022) we evaluate our approach on the Rotowire table-to-text dataset in a reverse direction, i.e., generate tables from text (Wiseman et al., 2017). Consequently, the complex tables reporting teams and player performance are generated given the game description. Results from Table 1 show that our T5+STable model can deliver an improvement over the *Linearized* T5 model on Rotowire Team. The fact that *Linearized* BART from Wu et al. (2022) outperforms our *Linearized* T5 baselines on Rotowire Team and Player datasets by 2.5 and 2.1 points, respectively, suggests that it has a better capacity as a backbone for this task. Several of the ablation studies from the next section were designed to shed light on this subject.

The results of our model (Table 1) demonstrate a significant improvement over the simple sequence-to-sequence generation of tables linearized as sequences on three out of five public datasets. As expected, it yields better results in cases where there is a considerable interdependency between values in a row and no clear, known upfront name distinguishes it from other rows. Note that, e.g., in Rotowire, it suffices to correlate all statistics with team or player name, which is always inferred first due to the employed linearization strategy. The order of columns being decoded is a hyperparameter in the case of linearization. In contrast, the power of STable comes from learning it from the data itself.

4 Ablation Studies

Models were trained three times with different random seeds on the Rotowire, DWIE, and PWC★ datasets. To reduce the computational cost, we relied on *base* variants of the models reported in Section 3 – please refer to Appendix D for detailed specifications and results. While results on a single dataset can be considered noisy, aggregation over a diverse set of them helps diminish the randomness’s impact and stabilize results on the new ones.

(1) Semi-templated Expansion. To compare our method of group prediction with a regression-free

Table 2: Results of studies (1), (2), (3), and (5). Modified models in relation to complete STable. See Appendix D for per-dataset results.

Model	Score	Change
Complete STable	62.9 ± 1.0	—
Semi-templated expansion	61.4 ± 0.1	-1.5 (1)
Fixed causal order	60.0 ± 0.4	-2.9 (2)
Decoding constraint		(3)
Column-by-column	62.4 ± 0.6	-0.5
Row-by-row	62.1 ± 0.6	-0.8
L→R and T→B	62.0 ± 0.5	-0.9
No distant rows	62.2 ± 0.5	-0.7
Sequential decoder bias only	3.9 ± 0.1	-59.0 (5)
Sequential and header bias	33.2 ± 0.3	-29.7

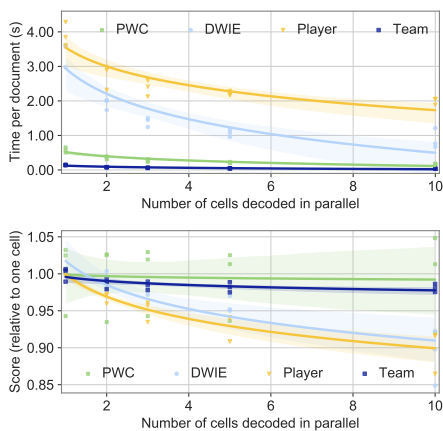


Figure 6: Results of decoding ablation (4). Three runs for 1, 2, 3, 5, and 10 cells decoded in parallel.

alternative, we allow the model to work in a semi-templated manner, where the template is infinite, and the decoding stops when the group with *NULL*-only tokens is generated. For this method, we add such a group at the bottom of the table during the training to comply with the inference. The model performance is significantly below the STable reference, suggesting explicit group number prediction superiority.

(2) Non-Permutative Training. To measure the importance of understanding the bidirectional contexts within the model, we abstain from permutation-based training in our study and choose the predefined factorization order. Here, a *fixed causal order* model that reads tables from left to right and from top to bottom is evaluated. This alternative is in line with text-to-table approach of Wu et al. (2022). As shown in Table 3, the lack of permutative training we introduced in Section 2 leads to significantly worse scores.

(3) Constrained Cell Order. Whereas the permutation-based training allows for great flexibility, the questions posed here are whether limiting the model’s ability to discover new cells can be of any value. Methods in this group assure either that the *column-by-column* constrained model predicts the whole column before decoding a new one, the *row-by-row* constrained model predicts the whole row before entering a new one, whereas *L→R* and *T→B* is a combination of both (ensures row-by-row inference from left to right). The *no distant rows* constraint forces the decoding to have empty cells only on the bottom of each column, thus avoiding skipping cells in the decoding while moving down.

As shown in Table 3, all but column-by-column constraint lead to a decreased scores. At the same time, the mentioned performs on par with STable’s model-guided inference (Section 2.4), and both are better than the method with left-to-right decoding order. These results suggest that (1) our method does not require constraining the decoding order, (2) it seems to implicitly incorporate the column-by-column constraint, and (3) it is helpful to be elastic w.r.t. decoding order within the column.

(4) Parallelization of Cell Decoding. As outlined in Section 2.4, one may allow multiple candidates to be kept in each decoding step to shorten the inference time while expecting the performance to degrade to some extent. Results of experiments that follow this observation are presented in Figure 6. One may notice that processing time varies across the considered datasets since it depends mainly on the input sequence length (ranging from $1k$ for Rotowire to $6k$ for PWC) and the sizes of the table to infer (we infer as many as 320 cells for the Player table). Parallelization of cell decoding significantly reduces the total per-document processing time — up to five times for DWIE in the conducted experiments. Interestingly, speed-up does not necessarily lead to a decrease in scores; e.g., in the case of the Team table, there is four times better processing time when ten cells are inferred at once, whereas the scores achieved by the model remain comparable. It can be attributed to the fact that there are almost no inter-cell dependencies and always only two rows to infer — one for each team playing. Since the performance changes w.r.t. this parameter is heavily data-dependent, its value should be obtained experimentally for each dataset separately. Additionally, we argue that it is beneficial to use large values to speed up the

train-time validation as it maintains a correlation with higher-scoring lower parameter values that can be employed during test-time inference.

(5) Tabular Attention Biases. In comparison with the initially introduced two relations (between cells and within cells), removing them and using only 1D global bias disrupts the permutation-based training as the model scores degrade since it cannot assign answers to correct columns. However, additional incorporation of the header name (by attending only to row with headers, $r_j = 0$ in Equation 4) leads to significant improvement, but it is still below the full model. Detailed analysis showed that the model could not benefit from 1D global bias, as (1) the distance between cells and header is too large for the first cells in the training since they are randomly chosen from any position within the table, and (2) a table itself is considerably bigger, as in permutation-based training we assumed that every cell in the table is generated, while for the linearized model, the headers are generated by the model, and a part of them can be skipped, thus reducing the size of the table. The consistent improvements on four datasets indicate that proposed tabular attention biases enhance table-modeling efforts.

5 Limitations

The state-of-the-art performance of STable is its foremost advantage, while the constraining factors come from different aspects. Of them, the generated sequence’s length seems to incur the most long-term cost during inference, while the increase in training time per example is a short-term obstacle. The underlying issue is that the full table context negatively influences the computational cost of the attention on the decoder side. This however is also the case for the family of encoder-decoder models generating the whole table such as these proposed by Wu et al. (2022) or Townsend et al. (2021). A possible solution here is a model with table context limited to the row and column a given table cell belongs to. Such a change would have a positive impact on the memory consumption in the decoder, as self-attention complexity decreases from $\mathcal{O}(NM)$ to $\mathcal{O}(N + M)$, where N, M denotes the number of rows and columns respectively. The exploitation of this optimization is an interesting future direction.

To navigate the intricacy of the order employed by the STable framework, we performed a system-

atical analysis that did not conclude in finding a visible decoding pattern that could be described formally beyond the observation already provided in Figure 5 and in constrained-decoding ablations. Studying the generation order in the context of data calls for designing a new explainability-related method, which is not in the scope of this work.

6 Summary

We equipped the encoder-decoder models consuming text (T5, T5 2D) and documents (TILT) with the capabilities to generate tables in a data-dependent order. Firstly, an aligned training procedure based on permuting factorization order of cells was presented, and secondly, the parallelizable decoding process that fills the table with values in a flexible and unconstrained order was proposed. The important design choices for both contributions were motivated by an extensive ablation study. The proposed STable framework demonstrates its high practical value by yielding state-of-the-art results on PWC[★] and outperforming linearized models on CORD and Rotowire Team datasets, as well as outperforming reference models on several confidential datasets. The highest gains due to the permutative training were accomplished on the PWC[★] dataset, where 4.0 points (26.8 → 30.8) amounts to 14.9% relative improvement, while the 8.8 point gain on Bank Statements (61.1 → 69.9) exceeds 14.4% relative improvement.

Acknowledgments

The Smart Growth Operational Programme partially supported this research under projects no. POIR.01.01.01-00-0877/19-00 (*A universal platform for robotic automation of processes requiring text comprehension, with a unique level of implementation and service automation*) and POIR.01.01.01-00-1624/20 (*Hiper-OCR - an innovative solution for information extraction from scanned documents*).

References

- Łukasz Borchmann, Michał Pietruszka, Tomasz Stanisławek, Dawid Jurkiewicz, Michał Turski, Karolina Szyndler, and Filip Graliński. 2021. [DUE: End-to-end document understanding benchmark](#). In *Proceedings of the Neural Information Processing Systems Track on Datasets and Benchmarks*, volume 1.
- Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan,

- Harrison Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri, Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastri, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Joshua Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. [Evaluating large language models trained on code](#). *CoRR*, abs/2107.03374.
- Tomasz Dwojak, Michał Pietruszka, Łukasz Borchmann, Jakub Chłędowski, and Filip Galiński. 2020. [From dataset recycling to multi-property extraction and beyond](#). In *Proceedings of the 24th Conference on Computational Natural Language Learning*, pages 641–651, Online. Association for Computational Linguistics.
- Marcin Kardas, Piotr Czapla, Pontus Stenetorp, Sebastian Ruder, Sebastian Riedel, Ross Taylor, and Robert Stojnic. 2020. [AxCell: Automatic extraction of results from machine learning papers](#). In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 8580–8594, Online. Association for Computational Linguistics.
- Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. [UNIFIEDQA: Crossing format boundaries with a single QA system](#). In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.
- Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2022. [Ocr-free document understanding transformer](#). In *Computer Vision – ECCV 2022*, pages 498–517, Cham. Springer Nature Switzerland.
- Ankit Kumar, Ozan Irsoy, Peter Ondruska, Mohit Iyyer, James Bradbury, Ishaan Gulrajani, Victor Zhong, Roman Paulus, and Richard Socher. 2016. [Ask me anything: Dynamic memory networks for natural language processing](#). In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48 of *Proceedings of Machine Learning Research*, pages 1378–1387, New York, New York, USA. PMLR.
- Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. [Natural questions: a benchmark for question answering research](#). *Transactions of the Association of Computational Linguistics*.
- Yaojie Lu, Hongyu Lin, Jin Xu, Xianpei Han, Jialong Tang, Annan Li, Le Sun, Meng Liao, and Shaoyi Chen. 2021. [Text2Event: Controllable sequence-to-structure generation for end-to-end event extraction](#). In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 2795–2806, Online. Association for Computational Linguistics.
- Bryan McCann, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher. 2018. [The natural language decathlon: Multitask learning as question answering](#). *CoRR*, abs/1806.08730.
- Seunghyun Park, Seung Shin, Bado Lee, Junyeop Lee, Jaeheung Surh, Minjoon Seo, and Hwalsuk Lee. 2019. [CORD: A consolidated receipt dataset for post-ocr parsing](#). In *Document Intelligence Workshop at NeurIPS*.
- Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. [Pytorch: An imperative style, high-performance deep learning library](#). In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.
- Rafał Powalski, Łukasz Borchmann, Dawid Jurkiewicz, Tomasz Dwojak, Michał Pietruszka, and Gabriela Pałka. 2021. [Going full-TILT boogie on document understanding with text-image-layout transformer](#). In *Document Analysis and Recognition – ICDAR 2021*, pages 732–747, Cham. Springer International Publishing.
- Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. [Exploring the limits of transfer learning with a unified text-to-text transformer](#). *JMLR*, 21(1).
- Kaiqiang Song, Bingqing Wang, Zhe Feng, and Fei Liu. 2021. [A new approach to overgenerating and scoring abstractive summaries](#).
- Mitchell Stern, William Chan, Jamie Kiros, and Jakob Uszkoreit. 2019. [Insertion transformer: Flexible sequence generation via insertion operations](#). In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5976–5985. PMLR.

- Benjamin Townsend, Eamon Ito-Fisher, Lily Zhang, and Madison May. 2021. [Doc2dict: Information extraction as text generation](#). *CoRR*, abs/2105.07510.
- Severine Verlinden, Klim Zaporozhets, Johannes Deleu, Thomas Demeester, and Chris Develder. 2021. [Injecting knowledge base information into end-to-end joint entity and relation extraction and coreference resolution](#). In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 1952–1957, Online. Association for Computational Linguistics.
- Fei Wang, Zhewei Xu, Pedro Szekely, and Muhao Chen. 2022. [Robust \(controlled\) table-to-text generation with structure-aware equivariance learning](#).
- Xing Wang, Zhaopeng Tu, Longyue Wang, and Shuming Shi. 2019. [Self-attention with structural position representations](#). In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 1403–1409, Hong Kong, China. Association for Computational Linguistics.
- Sam Wiseman, Stuart Shieber, and Alexander Rush. 2017. [Challenges in data-to-document generation](#). In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2253–2263, Copenhagen, Denmark. Association for Computational Linguistics.
- Lijun Wu, Xu Tan, Di He, Fei Tian, Tao Qin, Jianhuang Lai, and Tie-Yan Liu. 2018. [Beyond error propagation in neural machine translation: Characteristics of language also matter](#).
- Xueqing Wu, Jiacheng Zhang, and Hang Li. 2022. [Text-to-Table: A new way of information extraction](#). In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2518–2533, Dublin, Ireland. Association for Computational Linguistics.
- Zhilin Yang, Zihang Dai, Yiming Yang, Jaime Carbonell, Russ R Salakhutdinov, and Quoc V Le. 2019. [XLNet: Generalized autoregressive pretraining for language understanding](#). In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.
- Klim Zaporozhets, Johannes Deleu, Chris Develder, and Thomas Demeester. 2021. [DWIE: An entity-centric dataset for multi-task document-level information extraction](#). *Information Processing & Management*, 58(4):102563.
- Xu Zhong, Elaheh ShafieiBavani, and Antonio Jimeno Yepes. 2020. [Image-based table recognition: Data, model, and evaluation](#). In *Computer Vision – ECCV 2020*, pages 564–580, Cham. Springer International Publishing.

A Table Decoding Algorithm

The algorithm presented above operates on the output of the encoder model and reuses the cached encoded representations that are considered to be a part of the `DECODERMODEL` for brevity. Another important characteristic of the `DECODERMODEL` introduced for conciseness of the pseudocode is that it produces all cell tokens and handles the sequential text decoding on its own.

The decoding employs an `OUTERLOOP`, parametrized by the k parameter (denoting the parallelization of cell decoding) that progresses cell-by-cell, the `INNERLOOP` function that generates each cell that is yet to render, and `OUTERCRITERION` — a selection heuristics that determine which cell, from all the finalized in the inner loop, should be added to the outer loop. The `INNERCRITERION` is a heuristic we utilize that selects the cell with the maximum probability for its tokens’ predictions (Figure 5).

In the `INNERLOOP`, each cell is decoded until the special token determining the end of cell generation is placed. As the `INNERLOOP` generates each cell autoregressively and independently from other cells, the process can be treated as generating multiple concurrent threads of an answer and is well parallelizable. In the worst case, it takes as many steps as the number of tokens in the most extended cell.

After the selection by the `OUTERCRITERION` heuristic, the cell from the inner loop is inserted into the outer loop, and made visible to all other cells, while the cells that were not selected are to be reset and continuously generated in the future steps until they are chosen by the `OUTERCRITERION` heuristics.

B Negative Result: Prevention of Column Order Leakage

In the approach outlined in Section 2, the sequence of column labels \mathbf{c} , on which the likelihoods are conditioned, may leak additional unwanted information to the decoder. If the data in the document are indeed formatted as a table, and the order of labels in \mathbf{c} matches the column order, the model might learn to extract cells by location, instead of using the actual semantics of the cell label. However, during inference, while we know which entities we want to extract from the document, we are not given the order in which they appear, which

can be perceived as a serious train-inference discrepancy.

To remedy this problem, we tried to further modify the training objective (See Figure 7). Denote by \mathcal{C} the set of all non-empty sequences of distinct column labels. Instead of all the cells \mathbf{v} , we can predict only the cells \mathbf{v}_c corresponding to a sequence $\mathbf{c} \in \mathcal{C}$ of columns, in the order defined by the order of columns in \mathbf{c} . The expected log-likelihood over all $\mathbf{c} \in \mathcal{C}$ can be then expressed as

$$\log p_{\theta}(\mathbf{v}|\mathbf{h}) = \frac{1}{|\mathcal{C}|} \sum_{\mathbf{c} \in \mathcal{C}} \log p_{\theta}(\mathbf{v}_c|\mathbf{r}, \mathbf{c}), \quad (5)$$

where $p_{\theta}(\mathbf{v}_c|\mathbf{r}, \mathbf{c})$ decomposes according to the discussion in Section 2.

In practice, we found it to have no relevant impact on the training process. It did not lead to significant changes in evaluation scores when used in the supervised pretraining stage or on a downstream task. Consequently, we abandoned the idea and did not use it for any of the models reported in the paper. This study helps us state that the model learns the semantics of the cell labels without a need for regularization.

C Inner/Outer Loop Decision Criteria

The heuristic we test selects the cell in the outer loop based on the minimal or maximal inner score. Such inner score is calculated in three different ways: by taking the minimal, maximal, and mean of the token’s logits score. The results, presented in Table 3, point to the lesser importance of choosing the inner scoring method, while the choice of the outer loop heuristics impacts results more significantly. The former is the desired behavior since the algorithm we proposed in Section 2.4 is based on the assumption that it is beneficial to decode cells starting from those with the model’s highest confidence. On the other hand, as there is a significant variance depending on the dataset chosen (see Appendix D), these and other inference parameters can be subject to cost-efficient, task-specific hyperparameter optimization.

D Details of Experiments and Ablation Studies

All models were trained three times with different random seeds. We relied on *large* variants of the models for experiments in Table 1, and on *base* variants for the ablation studies. These are ana-

Algorithm 1 Table Decoding Algorithm of our proposal.

```
1: procedure OUTERLOOP( $k$ )
2:    $T \leftarrow 0_{n,m,l}$  ▷  $n \times m$  table with  $l$  padding tokens per cell
3:    $C \leftarrow 0_{n,m}$  ▷ current cell status (decoded or not)
4:   while SUM( $C$ ) <  $nm$  do ▷ while there is a cell to decode
5:      $T', L \leftarrow$  INNERLOOP( $T, C$ ) ▷ create complete table candidate  $T'$  and cell scores
6:      $\mathcal{B} \leftarrow$  OUTERCRITERION( $L$ ) ▷ sequence of coordinates sorted according to scores
7:     for  $c \leftarrow 1, k$  do ▷ for  $k$  best cells from  $T'$ 
8:        $i, j \leftarrow \mathcal{B}_c$  ▷ get coordinates
9:        $T_{i,j} \leftarrow T'_{i,j}$  ▷ ...copy values to table  $T$  accordingly
10:       $C_{i,j} \leftarrow 1$  ▷ ...and mark the appropriate cell as already decoded
11:    end for
12:  end while
13:  return  $T$ 
14: end procedure
15:
16: procedure INNERLOOP( $T, C$ )
17:    $L \leftarrow 0_{n,m}$  ▷ scores for each cell in  $n \times m$  table
18:    $T' \leftarrow T$  ▷ inner loop's table copy
19:   parfor  $i \leftarrow 1, n$  do ▷ for each table row
20:     parfor  $j \leftarrow 1, m$  do ▷ ...and each table cell processed in parallel
21:       if  $C_{i,j} = 0$  then ▷ ...if it was not decoded yet
22:          $s, t \leftarrow$  DECODERMODEL( $T, i, j$ ) ▷ produce cell tokens  $t$  and their scores  $s$ 
23:          $L_{i,j} \leftarrow$  INNERCRITERION( $s$ ) ▷ aggregate per-token scores into cell score
24:          $T'_{i,j} \leftarrow t$  ▷ update table copy
25:       end if
26:     end parfor
27:   end parfor
28:   return ( $T', L$ )
29: end procedure
30:
31: procedure INNERCRITERION( $s$ )
32:   /* Any  $\mathbb{R}^n \rightarrow \mathbb{R}$  function. STable assumes max, but we test other in the ablation studies. */
33: end procedure
34:
35: procedure OUTERCRITERION( $L$ )
36:   /* Some  $\mathbb{R}^{m \times n} \rightarrow (\mathbb{N} \times \mathbb{N})^{mn}$  function returning a permutation of indices of the input
37:   matrix  $L$ . STable assumes sort of matrix coordinates according to descending values of its
38:   elements, but we test other functions in the ablation studies. */
39: end procedure
40:
```

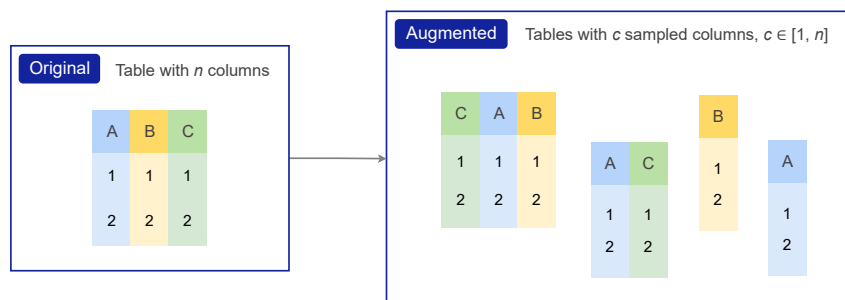


Figure 7: Change in training illustrated as augmentation of permuted sub-tables from the original table.

Table 3: Results of studies on decision criteria. Modified models in relation to complete STable. See Appendix D for per-dataset results.

Model	Score	Change
Complete STable	62.9 ± 1.0	—
Criteria (inner, outer)		
min max	61.7 ± 0.7	-1.2
mean max	62.7 ± 0.7	-0.2
mean min	60.8 ± 0.7	-2.1
min min	62.1 ± 0.4	-0.8
max min	61.2 ± 0.2	-1.7

lyzed in Table 3 given the average results over Rotowire, PWC[★], and DWIE datasets (see Table 4 for detailed scores).

Hyperparameters. We use task-independent hyperparameters that roughly follow these proposed by the authors of the T5 model for its finetuning, as during our initial experiments, they turned out to be a robust default (see Table 5).

Maximal input sequence lengths were chosen in such a way a fair comparison with reference models was ensured. In particular, we use T5+2D’s limit despite the fact one can achieve better results when consuming a more significant part of the input document. Similarly, the max number of updates follows the limit in reference models except for the DWIE dataset, where the state-of-the-art solution is based on the incomparable multi-step pipeline. See Table 6 for these task-specific details.

Software and hardware. All experiments and benchmarks were performed on DGX-A100 servers equipped with eight A100-SXM4-80GB GPUs that feature automatic mixed precision. Our models and references were implemented in PyTorch 1.8.0a0 (Paszke et al., 2019) with CUDA 11.4 and NVIDIA drivers 470.82.01.

E Business Datasets

Due to the sparsity of public benchmarks for complex information extraction, we decided to provide results on three confidential datasets. They assume, respectively, (1) the extraction of payments’ details from *Payment Stubs*, (2) *Recipe Composition* from documents provided by multinational snack and beverage corporation, as well as (3) account balances from *Bank Statements*. Their details are covered in the present section and Table 8.

Recipe Composition. The problem faced is extracting proprieties of food ingredients from confidential food manufacturer’s documentation. This dataset contains 165 annotated fragments from 55 documents, three pieces for each document, with annotations sourced from the corporation’s CRM system.

For each file, there are five tables to be extracted. The first one describes the ingredient’s physical and chemical parameters (i.e., parameter name, testing method, range of allowed values, unit of measurement, and testing method details). The second one describes sub-components of the ingredient (i.e., its quantity, name, allergens, ingredient function, and country of origin). The third table informs about the presence of allergens (e.g., their names and binary information about their presence). The last two tables contain a quantity of the allergens (e.g., names and their qualities) as sub-components and caused by contamination retrospectively.

The first table needs to be extracted from the first document fragment, the second table – from the second fragment, and the three last tables – from the third document fragment. Input documents feature tables and fulfilled forms, where properties are presented in the form of text or check-boxes.

The analysis of expected outputs shows a high level of variability concerning the factors of table length (1 to 60 rows) and answer type (either a

Table 4: Per-dataset results of studies (1), (2), (3), and (4). Modified models in relation to Complete STable.

Model	RW Player	RW Team	PWC★	DWIE	
Complete STable (reference)	82.7 ± 0.3	84.1 ± 0.7	27.5 ± 2.2	56.0 ± 1.4	
Semi-templated expansion	80.4 ± 0.5	84.1 ± 0.5	25.0 ± 0.8	56.1 ± 1.0	(1)
Fixed causal order	83.2 ± 0.4	84.3 ± 0.3	26.3 ± 1.6	46.5 ± 0.5	(2)
Decoding constraint					(3)
Column-by-column	82.5 ± 0.4	84.0 ± 0.5	28.4 ± 1.5	54.8 ± 0.8	
Row-by-row	80.2 ± 0.4	83.8 ± 0.4	27.6 ± 1.6	56.8 ± 0.8	
L→R and T→B	83.1 ± 0.5	84.1 ± 0.7	27.7 ± 1.8	53.2 ± 0.5	
No distant rows	82.7 ± 0.5	83.8 ± 0.6	28.1 ± 1.0	54.2 ± 1.2	
Decision criteria (inner × outer)					(4)
min max	81.9 ± 0.4	83.7 ± 0.5	26.5 ± 2.0	54.2 ± 0.8	
mean max	83.0 ± 0.3	83.8 ± 0.8	27.8 ± 1.4	56.1 ± 1.1	
mean min	81.2 ± 1.1	83.7 ± 0.6	26.4 ± 1.9	51.9 ± 0.5	
min min	82.8 ± 0.6	83.8 ± 0.5	27.6 ± 1.3	54.0 ± 0.5	
max min	82.3 ± 0.3	84.5 ± 1.0	20.7 ± 1.6	52.7 ± 0.4	
Sequential decoder bias only	0.3 ± 0.1	0.6 ± 0.3	14.1 ± 0.3	0.6 ± 0.1	(5)
Sequential and header bias	16.0 ± 0.4	45.1 ± 0.4	27.7 ± 2.0	44.2 ± 1.2	

Table 5: Task-independent hyperparameters used across all experiments.

Hparam	Dropout	Batch	Learning rate	Weight decay	Label smoothing	Optimizer
Value	.1	64	1e-3	1e-5	.1	AdamW

Table 6: Task-dependent hyperparameters and training details. (*) Length equal to the one consumed by the baseline model.

Dataset	Max steps		Max input length
	Ablation	Final	
PWC★	500	1,000	6,144*
Rotowire	3,000	8,000	1,024
CORD	—	36,000	1,024
DWIE	4,000	8,000	2,048
Recipe Composition	—	400	2600
Payment Stubs	—	—	—
Bank Statements	—	200	7000

binary value, number, complex chemical name, or a more extended description).

Payment Stubs. The second of our private datasets consists of 110 American payment stubs, i.e., documents obtained by an employee regarding the salary received.

We aim to extract employee and employer names, dates, and payment tables, where each row consists of payment type, hours worked, and payment amount. Since documents come from different companies, their layouts differ significantly.

Due to the straightforward form of information to be extracted, a single annotator annotated each document. We state these were annotated ethically by our paid co-workers.

Bank Statements. The last dataset consists of 131 annotated bank statements. The goal here is to extract bank and customer name, date of issue, and table of account balances (e.g., account number, balance at the beginning of the period, and balance at the end).

Data to be comprehended is partially presented in the document’s header and partially in multiple forms (each for one account).

Similar to the Payment Stubs dataset, documents here were issued by different banks and represent a broad spectrum of layouts. The annotation process was the same as for the Payment Stubs dataset.

F Adaptation to Table Structure Recognition Task

Our method by design does not generate the table header since we assume that the names of the datapoints to infer are given in advance. To tackle problems such as table structure recognition where the set of possible header values is not limited, one needs to slightly modify the proposed solution. However, we do not consider it a serious limitation as the required modification is relatively straightforward, and for the sake of completeness, we describe it below.

To adjust the proposed method to be applicable to the task of Table Structure Recognition, one must

Table 7: Detailed results of experiments on reversed Rotowire dataset. See Wu et al. (2022) for metrics’ specification.

	Row header F1			Column header F1			Non-header F1		
	Exact	Chrf	BERT	Exact	Chrf	BERT	Exact	Chrf	BERT
Team	94.9	95.2	97.8	88.9	85.8	88.7	84.7	85.6	90.3
Player	93.5	95.3	95.1	88.1	91.2	94.5	84.5	86.8	90.4

Table 8: Summary of the confidential datasets.

	Recipe Composition	Payment Stubs	Bank Statements
train documents		119	80
val documents		16	10
test documents		30	20
avg doc len (words)	0.6k	0.3k	1.3k
max doc len (words)	1.6k	2k	4, 9k
avg doc len (characters)	3.3k	2k	8.3k
max doc len (characters)	10k	14.2k	37.9k
properties total	64	11	10
properties in tables (tables columns)	64	4	4
properties outside of tables	0	7	6
mean number of table rows	12	5	2
max number of rows	60	15	5
mean length of cell (characters)	12	8	9
max length of cell (characters)	308	44	36

understand the differences in framing the problem between the tasks here.

Table Structure Recognition or Table Extraction aims to generate headers and the table content based on the document with the table provided explicitly. STable described in the main part of this paper can generate the table given any text and its position on pages. This capacity generalizes well to any input, including when the table is provided on the input. The difference is that the output form in STable assumes the headers are known upfront, while for Table Structure Recognition, inferring them is a part of the task. STable can achieve such capabilities to solve the Table Structure Recognition task by (1) adding a linear layer to predict the number of columns, (2) treating headers as the values to be inferred in the first row, (3) using dummy names of the columns, e.g., "first column," "second column," and (4) increasing the predicted number of rows by 1.

In this setup, the model will predict the number of columns and the number of rows, while the first row will represent the values of header names. The dummy headers will have to be removed during postprocessing, and the values in the first row should be treated as valid headers.

G Sample Input-Output Pairs

PWC★ (Borchmann et al., 2021). Input in the PWC★ consists of born-digital, multipage PDF files containing an article from the machine learning field. The expected output is a list of tuples describing achieved results on arbitrary datasets (see Figure 8).

CORD (Park et al., 2019). Input in the dataset is a single scanned or photographed receipt. From our point of view, the output here is twofold — there are simple data points that can be considered key-value pairs and data points that take the structured form of line items. We approach the problem as the generation of two tables from the document — one for each data kind (see Figure 9).

DWIE (Zaporojets et al., 2021). Input in the dataset is a plain-text article. The final goal is to extract the normed object, relation, and subject triples (though the original formulation assumes several intermediate stages). Triples are always complete (i.e., there are no NULL values, as we understand them (see Figure 10 for an example).

Reversed Rotowire (Wu et al., 2022). Input in the reversed Rotowire dataset, as reformulated by (Wu et al., 2022), is a plain-text sport news article. The task is to generate tables with team and

player statistics. The number of rows in the *Team* table is from zero (if no team is mentioned in the text) to two, whereas the number of rows in the *Player* is highly variable and content-dependent. Figure 11 present sample pair of document and tables to generate.

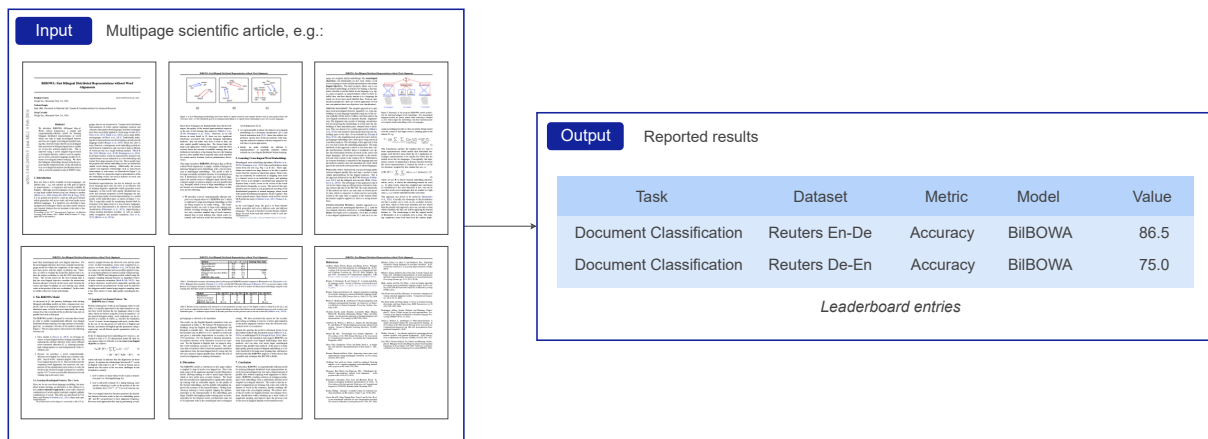


Figure 8: An example from PWC★ dataset considered in the document-to-table paradigm.

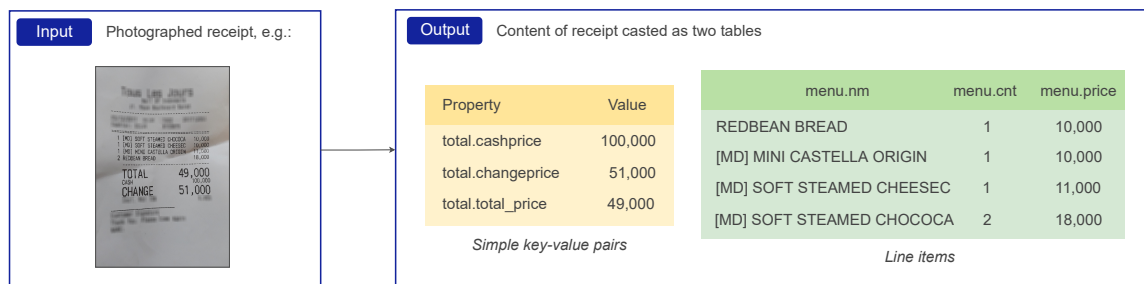


Figure 9: Sample document from CORD dataset and its expected output as interpreted in our approach.

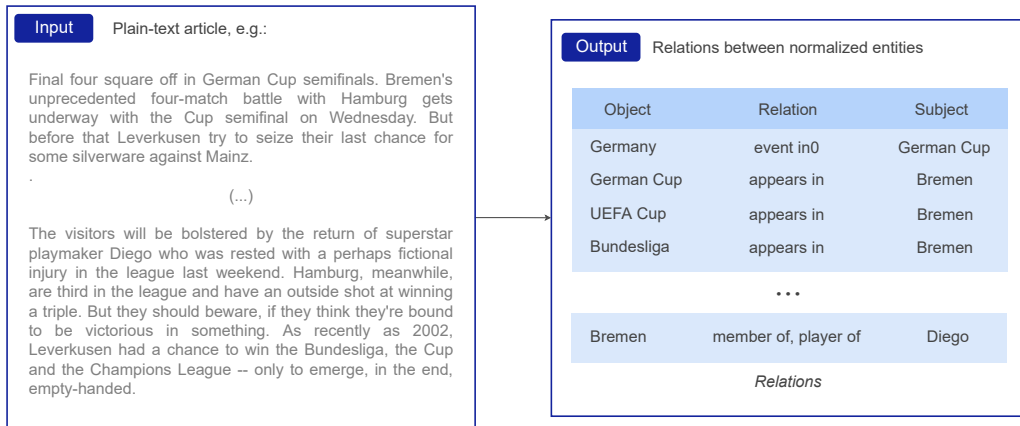


Figure 10: Sample input-output pair from the DWIE dataset. The table was shortened and consisted of 29 rows in our approach. Suppose multiple relations appear in the same direction between the pair of object-subject. In that case, we predict a list of them in a single cell, reducing the number of rows generated (see the example of the Bremen-Diego pair).

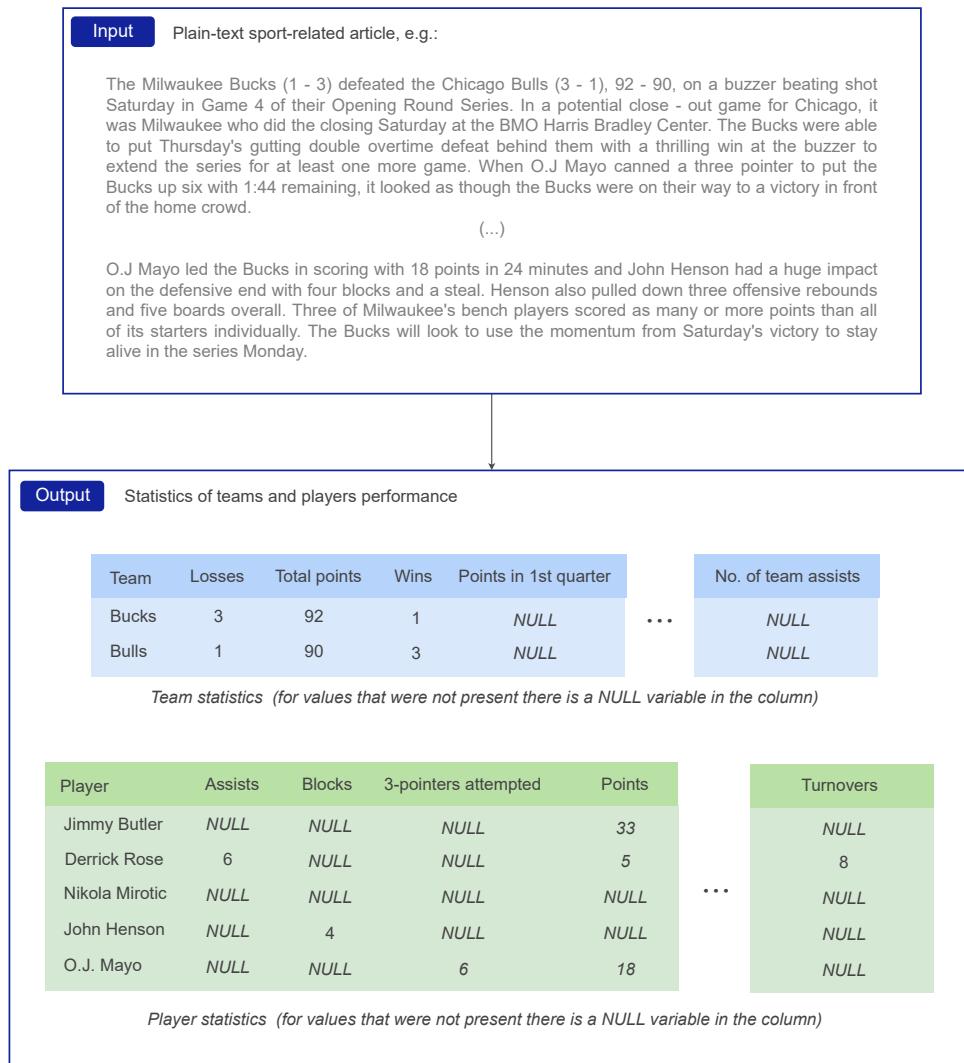


Figure 11: Input-output example from the reversed Rotowire dataset. We present shortened forms of tables than in real have 13 columns for Team and 20 columns for Player tables. Note that there is a NULL value in the column for values not present in the input text.



*The horse face emoji we feature is a part of Noto Emoji distributed under the Apache License 2.0.
Copyright by Google Inc. No animals were harmed in the making of this article.*

APPENDIX

A

Declarations of Contribution

Declaration

I hereby declare that the contribution to the following paper:
 Lukasz Borchmann, Michał Pietruszka, Tomasz Stanisławek, Dawid Jurkiewicz, Michał Turski, Karolina Szyndler, and Filip Graliński. "DUE: End-to-End Document Understanding Benchmark". In: *Under review in NeurIPS 2021*. 2021. URL: <https://openreview.net/forum?id=rHs2FvJGDK>
 is correctly characterized in the table below (* denotes equal contributions).

Contributor	Description of main tasks
Lukasz Borchmann*	<ul style="list-style-type: none"> - conceptualization and methodology (participated in regular discussions) - methodology of the considered datasets for DUE benchmark - implementation of baselines - create DUE benchmark webpage - create scripts for evaluation - convert documents from TabFact and WTQ datasets into pdf files - result analysis - writing the paper - organizing and controlling the process of human annotation
Michał Pietruszka*	<ul style="list-style-type: none"> - conceptualization and methodology (participated in regular discussions) - methodology and preparation list of the considered datasets for DUE benchmark - implementation of baselines - preparation of datasets (DocVQA, InfographicsVQA, WikiTableQuestions, PWC) - preparing code, models and datasets for final release - result analysis - writing the paper - organizing and controlling the process of human annotation
Tomasz Stanisławek*	<ul style="list-style-type: none"> - conceptualization and methodology (participated in regular discussions) - methodology and preparation list of the considered datasets for DUE benchmark - prepare schema for storing benchmark datasets in unified data format - preparation of datasets (Kleister Charity, DeepForm, TabFact) - curation of PWC and DeepForm datasets - methodology for creation of the diagnostic subsets - result analysis - improved the first version of the paper / edition of the manuscript - organizing and controlling the process of human annotation
Dawid Jurkiewicz	<ul style="list-style-type: none"> - participated in regular discussions - implementation of baselines - significantly improved results of the baselines (hyper-param search for it) - performing experiments - preparing code and models for final release - edition of the paper
Michał Turski	<ul style="list-style-type: none"> - methodology and preparation of the diagnostic subsets - organizing and controlling the process of human annotation - controlling the process of measuring human performance where it was required (PWC, DeepForm, WTQ) - edition of the paper
Karolina Szyndler	<ul style="list-style-type: none"> - improving schema for storing benchmark datasets in unified data format - code for reading datasets by the baselines
Filip Graliński	<ul style="list-style-type: none"> - participated in regular discussions - edition of the paper

Lukasz Borchmann

Michał Pietruszka

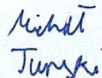
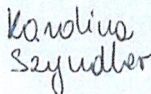
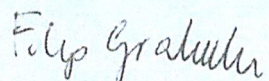
Tomasz Stanisławek

Dawid Jurkiewicz

Michał Turski

Karolina Szyndler

Filip Graliński

Declaration

I hereby declare that the contribution to the following paper:

Lukasz Borchmann, Michał Pietruszka, Tomasz Stanisławek, Dawid Jurkiewicz, Michał Turski, Karolina Szyndler, and Filip Galiński. “DUE: End-to-End Document Understanding Benchmark”. In: *Under review in NeurIPS 2021*. 2021. URL: <https://openreview.net/forum?id=rNs2FvJGDK> is correctly characterized in the table below (* denotes equal contributions).

Contributor	Description of main tasks
Lukasz Borchmann*	<ul style="list-style-type: none"> - conceptualization and methodology (participated in regular discussions) - methodology of the considered datasets for DUE benchmark - implementation of baselines - create DUE benchmark webpage - create scripts for evaluation - convert documents from TabFact and WTQ datasets into pdf files - result analysis - writing the paper - organizing and controlling the process of human annotation
Michał Pietruszka*	<ul style="list-style-type: none"> - conceptualization and methodology (participated in regular discussions) - methodology and preparation list of the considered datasets for DUE benchmark - implementation of baselines - preparation of datasets (DocVQA, InfographicsVQA, WikiTableQuestions, PWC) - preparing code, models and datasets for final release - result analysis - writing the paper - organizing and controlling the process of human annotation
Tomasz Stanisławek*	<ul style="list-style-type: none"> - conceptualization and methodology (participated in regular discussions) - methodology and preparation list of the considered datasets for DUE benchmark - prepare schema for storing benchmark datasets in unified data format - preparation of datasets (Kleister Charity, DeepForm, TabFact) - curation of PWC and DeepForm datasets - methodology for creation of the diagnostic subsets - result analysis - improved the first version of the paper / edition of the manuscript - organizing and controlling the process of human annotation
Dawid Jurkiewicz	<ul style="list-style-type: none"> - participated in regular discussions - implementation of baselines - significantly improved results of the baselines (hyper-param search for it) - performing experiments - preparing code and models for final release - edition of the paper
Michał Turski	<ul style="list-style-type: none"> - methodology and preparation of the diagnostic subsets - organizing and controlling the process of human annotation - controlling the process of measuring human performance where it was required (PWC, DeepForm, WTQ) - edition of the paper
Karolina Szyndler	<ul style="list-style-type: none"> - improving schema for storing benchmark datasets in unified data format - code for reading datasets by the baselines
Filip Galiński	<ul style="list-style-type: none"> - participated in regular discussions - edition of the paper

Lukasz Borchmann

Michał Pietruszka

Tomasz Stanisławek

Dawid Jurkiewicz

Michał Turski

Karolina Szyndler

Filip Galiński

Dawid Jurkiewicz

Warsaw, April 4, 2024

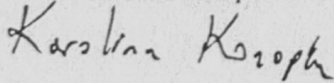
Declaration of Contribution

I hereby declare that the contribution to the following paper:

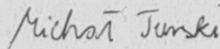
Karolina Konopka, Michał Turski, and Filip Graliński, *Arxiv Tables: Document Understanding Challenge Linking Texts and Tables*, in *Document Analysis and Recognition – ICDAR 2023 Workshops, 2023* is correctly characterized in the table below.

Contributor	Description of main tasks
Karolina Konopka	Conceptualization and methodology of creating the challenge, design and implementation of a tool creating the dataset, exploration of the dataset, evaluation procedure and platform, writing the paper
Michał Turski	Conceptualization and implementation of proposed baseline solutions, running experiments, writing and editing the paper
Filip Graliński	Conceptualization and methodology of creating the challenge, evaluation procedure and platform, description of the related works

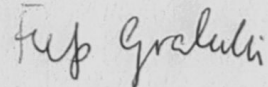
Karolina Konopka



Michał Turski



Filip Graliński



Warsaw, April 4, 2024

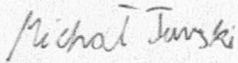
Declaration of Contribution

I hereby declare that the contribution to the following paper:

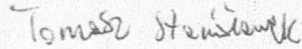
Michał Turski, Tomasz Stanisławek, Karol Kaczmarek, Paweł Dyda, and Filip Galiński, *Cpdf: Building a High Quality Corpus for Visually Rich Documents from Web Crawl Data*, in *Document Analysis and Recognition – ICDAR 2023*, 2023 is correctly characterized in the table below.

Contributor	Description of main tasks
Michał Turski	Conceptualization and methodology, design and implementation of the tool for managing the corpus, statistical analyses and exploration of the corpus, running proposed pipeline and sharing the data, writing and editing the paper, project leadership
Tomasz Stanisławek	Design and implementation of the tool for managing the corpus, comparative studies of OCR and language detection tools, born-digital detection tool, writing the paper
Karol Kaczmarek	Implementation tools for link extraction, URL-based language detection, and downloading files
Paweł Dyda	Design and implementation tool for PDF's metadata extraction, born-digital detection tool
Filip Galiński	The idea of a new corpus of PDF files, design and implementation tools for link extraction, URL-based language detection, and downloading files, writing the paper

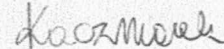
Michał Turski



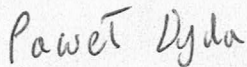
Tomasz Stanisławek



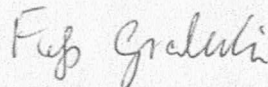
Karol Kaczmarek



Paweł Dyda



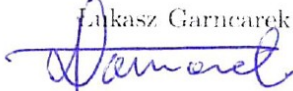
Filip Galiński




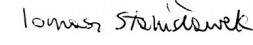
Declaration


I hereby declare that the contribution to the following paper:
 Lukasz Garncarek, Rafał Powalski, Tomasz Stanisławek, Bartosz Topolski, Piotr Halama, Michał Turski, and Filip Graliński. “LAMBERT: Layout-Aware Language Modeling for Information Extraction”. In: *Document Analysis and Recognition – ICDAR 2021*. Cham: Springer International Publishing, 2021, pp. 532–547
 is correctly characterized in the table below (* denotes equal contributions).

Contributor	Description of main tasks
Lukasz Garncarek*	<ul style="list-style-type: none"> – conceptualization and methodology (participated in regular team discussions) – implemented basic framework for models with context embeddings – result analysis for ablation study section – wrote the paper
Rafał Powalski*	<ul style="list-style-type: none"> – conceptualization and methodology (participated in regular team discussions) – implemented basic framework for models with context embeddings – proposed and implemented relative embeddings – prepared SROIE dataset and run initial experiments – edition of the paper
Tomasz Stanisławek*	<ul style="list-style-type: none"> – initialize the whole idea of using BERT based model to Key Information Extraction task – conceptualization and methodology (participated in regular team discussions) – prepared datasets (for model training and model evaluation of downstream tasks) – implemented scripts for automation of experiments on downstream tasks – run final experiments on NDA, Kleister and SROIE datasets – propose, design and do result analysis for the ablation study section – edition of the paper
Bartosz Topolski*	<ul style="list-style-type: none"> – conceptualization and methodology (participated in regular team discussions) – implemented scripts for automation of experiments on downstream tasks – run initial experiments on downstream tasks – significantly improved framework for models with context embeddings – edition of the paper
Piotr Halama	<ul style="list-style-type: none"> – prepare scripts for running experiments on LayoutLM – run all experiments for CORD dataset – edition of the paper
Michał Turski	<ul style="list-style-type: none"> – create final dataset for model training with methodology for filtering out non business/legal documents – implementation of training flow for training LAMBERT models – edition of the paper
Filip Graliński	<ul style="list-style-type: none"> – supervised team (participated in regular team discussions) – evaluation methodology – review of the paper

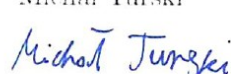
Lukasz Garncarek


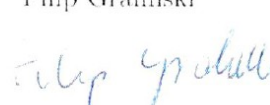
Rafał Powalski


Tomasz Stanisławek


Bartosz Topolski


Piotr Halama


Michał Turski


Filip Graliński


Warsaw, February 29, 2024

Declaration of Contribution

I hereby declare that the contribution to the following paper:

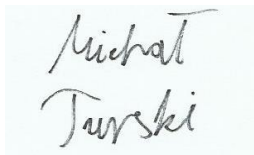
Michał Pietruszka, Michał Turski, Łukasz Borchmann, Tomasz Dwojak, Gabriela Nowakowska, Karolina Szyndler, Dawid Jurkiewicz, and Łukasz Garncarek, *STable: Table Generation Framework for Encoder-Decoder Models*, Proceedings of the 18th Conference of the European Chapter of the Association for Computational Linguistics: EACL 2024, 2024 is correctly characterized in the table below (* denote group of equal contribution).

Contributor	Description of main tasks
Michał Pietruszka*	Conceptualization and methodology of the research work, idea behind the solution as a whole, novel algorithm development and implementation, experimental design and implementation, real-world applications, writing a paper
Michał Turski*	Conceptualization and methodology of the research work, idea behind pre-training, preparation of domain-specific pre-training dataset, data preprocessing and postprocessing, baselines implementation, running pre-training, experiments, and ablation studies, error analysis, writing a paper, project leadership
Łukasz Borchmann*	Conceptualization and methodology of the research work, major participation in brainstorming that led to the final solution, comparative studies, theoretical analysis, design and preparation of experiments with internal datasets, running experiments, error analysis, writing a paper, designing and conducting ablation studies, improvement of the original implementation.
Tomasz Dwojak	Baselines implementation, running experiments, participation in discussions and brainstorming, team management, preparing data model for experiments
Gabriela Nowakowska	Review and preparation of the datasets, baselines implementation, running experiments, participation in discussions and brainstorming, editing the manuscript in its initial version
Karolina Szyndler	Baselines implementation, experiments preparation, participation in discussions and brainstorming
Dawid Jurkiewicz	2D tabular embeddings (co-invention with Michał Pietruszka), conceptual work
Łukasz Garncarek	Baselines implementation, participation in discussions, and brainstorming

Michał Pietruszka



Michał Turski



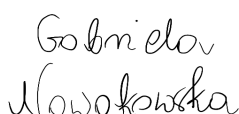
Łukasz Borchmann



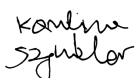
Tomasz Dwojak



Gabriela Nowakowska



Karolina Szyndler



Dawid Jurkiewicz



Łukasz Garncarek



B

Awards, Patents, and Projects

B.1 IAPR/ICDAR 2021 Best Industry Related Paper Award



B.2 Encoder-decoder transformer for table generation patent

Title: Encoder-decoder transformer for table generation

Patent number: US20230297554A1

Inventors: Łukasz Konrad Borchmann, Tomasz Dwojak, Łukasz Sławomir Garncarek, Dawid Andrzej Jurkiewicz, Michał Waldemar Pietruszka, Gabriela Klaudia Pałka, Karolina Szyndler, Michał Turski

Patent granted: 02.01.2024

Abstract: Systems and methods for generating tables are provided. The systems and methods perform operations comprising accessing a text document comprising a plurality of strings; processing the text document

by a machine learning model to generate a table comprising a plurality of entries that organizes the plurality of strings into rows and columns over a plurality of iterations; and at each of the plurality of iterations, estimating by the machine learning model a first value of a first entry of the plurality of entries based on a second value of a second entry of the plurality of entries that has been determined in a prior iteration.

B.3 Projects

I actively participated in the following research projects

- ▶ Uniwersalna platforma robotyzacji procesów wymagających rozumienia tekstu o unikalnym poziomie automatyzacji wdrożenia i obsługi (POIR.01.01.01-00-0877/19)
- ▶ Hiper-OCR – innowacyjne rozwiązanie do ekstrakcji informacji z dokumentów skanowanych (POIR.01.01.01-00-1624/20)