UNIVERSITY OF BONN

BACHELORARBEIT

Content Extraction and Subsequent Statistical Analysis of Scientific Papers in PDF Format

---

Lysander Lenzen

lenzen.lysander@gmail.com

First Examiner	*Second Examiner*	*Supervisors*
Prof. Dr. Matthew Smith	Prof. Dr. Michael Meier	Anna-Marie Ortloff, Florin Martius

Institute of Computer Science 4

Behavioral Security Group

12.04.2024

ABSTRACT

Statistical analysis of research papers is crucial for validating reported results, but it often becomes a tedious manual process due to the PDF format in which these papers are typically available. Kamleh [21] attempted to address this by developing a method using Regular Expressions (RegEx) to automate the extraction of statistical values from text. While her approach was effective, it lacked flexibility in detecting unobserved notations and did not account for values reported within tables. This thesis proposes an alternative method that utilizes Optical Character Recognition (OCR) technology to extract content from these papers, which is then categorized into tabular and non-tabular data, so it can be further analyzed with a Large Language Model (LLM). The new approach was tested on publications from the Symposium on Usable Privacy and Security (SOUPS) and the Conference on Human Factors in Computing Systems (CHI), achieving mean F_1-Scores of 0.841 and 0.827 for text-based value extraction, and 0.623 and 0.582 for table-based value extraction, respectively. These results indicate performance that either improves upon or is comparable to existing techniques, yet still shows potential for further refinement. In contrast, another method developed in this thesis, which employs HTML files, reached higher mean F_1-Scores of 0.938 for text extraction and 0.924 for table extraction. Despite the superior performance of the HTML-based approach, the PDF-based method remains highly valuable due to the prevalent use of PDFs for publishing research papers. This highlights the continuous need for advancements in PDF data extraction technologies and emphasizes the importance of enhancing these methods to improve their utility and accuracy in academic research.

Declaration of Authorship

I hereby declare that I am the sole author of this bachelorarbeit and that I have not used any sources other than those listed in the bibliography and identified as references. I further declare that I have not submitted this thesis at any other institution in order to obtain a degree.

Bonn, April 26th, 2024

(Place, Date)

(Signature)

Table of Contents

List of Figures

LIST OF TABLES

LIST OF LISTINGS

1 Introduction

Since its inception in 1993, the PDF format has become the de facto standard for document dissemination across business, government, and particularly academia. Consequently, a vast majority of scientific papers are submitted as PDF files.

Analyzing research results presented in these papers is often a laborious manual process. The PDF format primarily focuses on visual consistency across different systems, which complicates automatic content extraction because its encoding emphasizes visual layout rather than structural information. This bachelor thesis proposes an automated method to extract statistical values such as p-values and effect sizes from PDF-only research papers, significantly reducing the time required for manual analysis.

Kamleh [21] previously addressed this issue by developing a method to extract sentences containing statistical values using a Regular Expressions (RegEx) approach, achieving an extraction success rate of 89.8%. However, her method failed to isolate the values within these sentences and could not extract data from tabular structures.

This thesis aims to overcome the limitations noted in Kamleh's work by developing techniques for accurately capturing values from both textual and tabular structures and isolating these values for further analysis. The challenges identified and addressed in this thesis include:

1. Extracting textual content from PDF files.
2. Extracting tabular content from PDF files.
3. Isolating statistical parameters from both textual and tabular data.

To tackle these challenges, this thesis explores and integrates approaches used in existing literature into a comprehensive method. This involves the use of Optical Character Recognition (OCR) technology for data extraction, which has proven especially effective in handling tabular data due to its superior ability to preserve structural integrity, crucial for accurate table reconstruction. A novel approach was also developed to distinguish between tabular and non-tabular content by analyzing text around table captions, identified using RegEx. Additionally, a Large Language Model (LLM) is employed to extract statistical values from the retrieved content. LLMs are particularly suited to this task as they leverage not only syntax but also contextual understanding, providing a unique advantage in accurately detecting and isolating these values.

To evaluate this thesis's method, the F_1-Score was selected because it equally balances recall and precision, as both are of equal interest to this thesis. When tested, the method's table extraction capabilities surpassed those of popular non-proprietary alternatives, Tabula and Camelot. The lowest mean F_1-Score achieved by this thesis's method was 0.659, significantly higher than Tabula's 0.44 and Camelot's 0.41. For value extraction, the lowest mean F_1-Score from tables was 0.582, which improves to 0.878 when considering only successfully extracted tables. For text-based extraction, the lowest mean F_1-Score recorded was 0.827. Although these results do not show a significant improvement in extraction rates over Kamleh's approach, this thesis's method distinguishes itself by isolating found statistical values, thereby enabling more straightforward further processing. Furthermore, an alternative HTML-based method developed in this thesis demonstrated even higher mean F_1-Scores of 0.924 for table-based and 0.938 for text-based value extraction, underscoring the potential for further enhancements if issues like OCR induced typographical errors are effectively addressed.

By developing this automated method, this thesis contributes to reducing the manual labor involved in analyzing research results, thereby streamlining data analysis processes in academic research.

2 Related Work

The objective of this chapter is to review existing literature that has addressed tasks similar to the primary challenges outlined in the introduction. This review enables this thesis to gain insights into tools and methodologies that could be effective in tackling the central issues of this thesis.

2.1 Text extraction and statistical content search of PDF files

As already stated in the introductory chapter (Chapter 1), Kamleh [21] wrote her thesis aiming to address the same problem as this thesis does. Consequently, she faced similar challenges, providing a solid foundation of research to build upon.

To solve the issue of content extraction, Kamleh [21] utilized and tested three different python packages, dedicated to this exact purpose: pdfplumber[1], PyPDF2[2] and pdfminer[3]. These packages were tested on a set of 20 papers sampled from the Symposium on Usable Privacy and Security (SOUPS) in 2021 and 2022. During testing she found that while all of the packages struggled with the presentation and/or extraction of tabular data, pdfminer had very few problems in other tested areas and proved to be the most consistent in transforming PDF text content into a machine-readable format.

The issue of statistical value extraction was solved by Kamleh [21] using a RegEx approach. To achieve this, she segmented the previously extracted text content into sentences, allowing each one to be searched for statistical content. This search for statistical content was conducted using a list of RegEx patterns containing statistical terms, value representations and test names. To compile this list, she employed a Natural Language Processing (NLP) approach, which searched all SOUPS Papers from the years 2021 and 2022 for statistical information to serve as a basis for the RegEx patterns.

To test the performance of her method, Kamleh [21] applied it to extract sentences containing statistical content from all SOUPS papers published between 2010 and 2020. She calculated the extraction rate by manually examining one paper from each year. She found that her method was able to extract 89.8% of sentences containing statistical content. She aimed to improve this performance for future use by adding new RegEx patterns for the statistical terms she missed.

2.2 Table extraction from PDF files

Given that Kamleh [21] encountered difficulties in extracting tabular content structurally intact within the scope of her project and therefore was unable to extract statistical content within these tables, this thesis identified this challenge as particularly intriguing and recognized significant potential for improvement.

One of the earliest open-source projects aiming to extract tabular content from PDFs dates back to 2004 and was developed by Yildiz for her Master's Thesis [45]. While she continued working on the project in the following year (Yildiz et al. [46]), the performance of her method reportedly lags behind more modern ones [8].

This is demonstrated by Corrêa et al. [8], who conducted a survey of scientific papers that utilized table extraction methods. They evaluated these tools based on the reported results from papers that employed them, utilizing various criteria regarding 'ease of use' and 'output

[1] https://pypi.org/project/pdfplumber/
[2] https://pypi.org/project/PyPDF2/
[3] https://pypi.org/project/pdfminer/

results'. Their evaluation revealed that the method by Yildiz et al. [46] severely underperformed compared to other methods, receiving a rating of 1.6 out of 5.0. While most of the other analyzed tools were proprietary products, they also assessed Tabula[4], another open-source project, which received a rating of 4.0 out of 5.0. Despite receiving a lower rating compared to its proprietary counterparts, the authors described it as "[...] an interesting alternative." and stated that "[...] Tabula showed excellent output results compared to commercial [tools]."
A benchmark study conducted by Meuschke et al. [26] assessed various tools across a range of PDF extraction tasks, including the extraction of tabular data. This study utilized academic documents for testing, thereby enhancing the significance of their findings for the purposes of this thesis. The results indicated that the most successful table extraction tools, ranked in descending order of effectiveness, were: Adobe Extract[5] ($F_1 = 0.47$), Camelot[6] ($F_1 = 0.30$), and Tabula ($F_1 = 0.28$). All of the tested tools exhibited notably low F_1 scores, with the study also highlighting specific issues with the non-proprietary tools Camelot and Tabula, noting that: "Both Camelot and Tabula incorrectly treat two-column articles as tables [...]".

This suggested a potential necessity for the development of a novel table extraction approach as an integral component of this thesis, specifically designed to address the challenges associated with extracting tables from two-column articles, as these are fairly common among research papers.

2.3 Extraction of statistical content

Despite achieving a high extraction rate using her RegEx approach, Kamleh [21] identified issues with this method herself. For example, she discovered nine different possible ways just for presenting p-values in SOUPS papers from the years 2021 and 2022. This highly unregulated and non-standardized way of statistical reporting demanded a significant number of RegEx patterns to capture just one of the many statistical values of interest (See Listing 2.1).

The overwhelming amount of RegEx patterns needed to extract all statistical content found in just SOUPS papers, warrants a more flexible approach, one that is able to extract statistical content regardless of its presentation in text.
Pires [34] employed such a context-aware approach in his work. He tested a pre-trained LLM, specifically OpenAI's Assistant API[7], to conduct statistical analysis tasks. For this, he provided the Assistant API with a CSV file and a task (e.g., conducting a t-test), demonstrating its capability to extract and analyze data from files. He found its performance to be promising and described it as a powerful tool to assist in statistical analysis, stating that: "[...] once they have a basics on statistics, they can perform statistical analysis on their datasets effortless [...]".

While Pires [34] only tested the Assistant API using CSV files, it seemed a worthy endeavor to also test its extraction capabilities on TXT files, which are relevant for extraction from non-tabular data.

[4] https://tabula.technology/
[5] https://developer.adobe.com/document-services/apis/pdf-extract/
[6] https://github.com/camelot-dev/camelot
[7] https://platform.openai.com/docs/assistants/overview

Listing 2.1: Python RegEx patterns developed by Kamleh [21] for the extraction of p-values from textual data.

```
# p-values
        (r"␣\.?,?;?\(?\s*p\s*[<=>≥≤]\s*\.\d+\s*\)?\s*\.?,?;?", 'p-value'),
        (r"␣\.?,?;?\(?\s*p\s*[<=>≥≤]\s*1\s*\)?\s*\.?,?;?", 'p-value'),
        (r"␣\.?,?;?\(?\s*p\s*[<=>≥≤]\s*0\.\d+\s*\)?\s*\.?,?;?", 'p-value'),
        (r"␣\(?p\s*=\s*\d+\.\d+e", 'p-value'),
        (r"[Pp][--]?[Vv]alues?", 'p-value␣Context'),
        (r"␣\.?,?;?\(?0?\.\d+␣[Pp]\s*[--]?\s*[Vv]alues?\s*\)?␣\.?,?;?␣", 'p-value'),
        (r"[Pp][--]?[Vv]alues?\s*[=<>≥≤]\s*\.\d+", 'p-value'),
        (r"[Pp]\s*\(.*\)\s*[=<>≥≤]\s*\.\d+", 'p-value'),
        (r"[Pp]\s*\(.*\)\s*[=<>≥≤]\s*0\.\d+", 'p-value'),
        (r"[Pp][--]?[Vv]alues?\s*[=<>≥≤]\s*\d+\.\d+", 'p-value'),
        (r"[Pp][--]?[Vv]alues?␣.*\s*smaller␣than␣\.\d+", 'p-value'),
        (r"[Pp][--]?[Vv]alues?␣.*\s*smaller␣than␣0\.\d+", 'p-value'),
        (r"[Pp][--]?[Vv]alues?␣.*\s*bigger␣than␣\.\d+", 'p-value'),
        (r"[Pp][--]?[Vv]alues?␣.*\s*bigger␣than␣0\.\d+", 'p-value'),
        (r"[Pp][--]?[Vv]alue␣less␣than␣\d+\.\d+", 'p-value␣Value'),
        (r"[Pp][--]?[Vv]alue␣less␣than␣\.\d+", 'p-value␣Value'),
        (r"[Pp][--]?[Vv]alues?␣.*\s*are␣.*\s*0\.\d+\s*", 'p-value'),
        (r"[Pp][--]?[Vv]alues?␣.*\s*are␣.*\s*.\d+\s*", 'p-value'),
        (r"[Pp][--]?[Vv]alues?␣of␣0\.\d+\s*", 'p-value'),
        (r"[Pp][--]?[Vv]alues?␣of␣.\d+\s*", 'p-value'),
        (r"[Pp][--]?\s*[Vv]alues?␣range?i?n?g␣.*\s*0\.\d+␣to␣0\.\d+", 'p-value␣Range'),
        (r"[Pp][--]?\s*[Vv]alues?␣range?i?n?g␣.*\s*\.\d+␣to␣\.\d+", 'p-value␣Range'),
        (r"corr?[--]?\s*p\s*[<=>≥≤]\s*\.\d+", 'corrected␣p-value'),
        (r"corr?[--]?\s*p\s*[<=>≥≤]\s*0\.\d+", 'corrected␣p-value'),
        (r"corrected␣[Pp][--]?[Vv]alues?", 'corrected␣p-value'),
```

3 METHODOLOGY

This chapter evaluates the effectiveness of the methodologies discussed in the related work chapter (Chapter 2) by examining their performance in executing tasks specific to this thesis. The evaluations were conducted using research papers from the field of Human-Computer Interaction (HCI), a domain previously used to test Kamleh's approach [21] and recommended by this thesis's supervisors for analyzing papers with the proposed methods.

This section will detail the selection process for the chosen methods by demonstrating their superior performance and suitability for achieving the objectives of this thesis compared to alternative approaches. It will also explain how each chosen method contributes to the overall workflow and conclude by visualizing the combined processes, highlighting the integration and synergy between the chosen methods.

3.1 EVALUATING TABLE EXTRACTION METHODS

In the initial development stages of this thesis, priority was given to evaluating the researched tabular extraction tools due to the absence of a viable method identified by Kamleh [21] to tackle this task.

Furthermore, if these tools yield subpar performance akin to the findings of Meuschke et al. [26], the necessity for a novel approach would become evident at an earlier stage. This would enable the allocation of additional time for the development of such an approach.

The evaluation involved five papers from the SOUPS published in 2023. Specifically, these papers were authored by Akter et al. [3], Hazazi et al. [15], Kelley et al. [22], Kersten et al. [23], and Liu et al. [25]. These selections were deliberate, as they covered a diverse range of table complexities and styles.

The tools evaluated were Camelot and Tabula, noted as the non-proprietary tools with the most promising performance, based on the benchmark conducted by Meuschke et al. [26]. These tools were tested using their default settings to avoid introducing bias. However, should one of them show promise, adjusting the settings would be the subsequent step for further optimization.

The extraction results, observed when applying these tools to the aforementioned papers, reveal a rather dismal performance, as depicted in Table 3.1. Even if this thesis generously considers the partially extracted tables as true positives, the tools still achieve F_1-Scores of approximately 0.41 (Camelot) and 0.44 (Tabula). Camelot's performance is hindered by a high number of false negatives, whereas Tabula exhibits a high number of false positives. Although Tabula may seem viable, given the possibility to filter out a significant portion of false positives, it is important to note that more than half of its extracted tables are only partially extracted. Consequently, further processing would be necessary, but its low F_1-Score does not justify such additional efforts.

It is worth noting that while Tabula exhibited the reported issue of incorrectly identifying text sections in two-column papers as tables [26], Camelot did not and instead its falsely extracted content mostly contained empty entries.

Furthermore, the tables extracted in full by Tabula were primarily those spanning the entire width of the paper, further indicating that the issue does lie in the two-column layout of the analyzed papers. Thus, while not compatible with the goals of this thesis, Tabula itself does seem to be a promising extraction tool, provided that the layout issue is not of concern.

For this thesis, however, a novel approach is necessary to address the challenge of table extraction, which will be further elaborated on later in this chapter.

Table 3.1: Extraction results by Camelot and Tabula

	Camelot	Tabula
Fully Extracted Tables	8	14
Partially Extracted Tables*	4	18
Tables Not Extracted	26	6
Falsely Extracted Content	9	75
Total Number of Tables Present	38	

* Tables with partially extracted content or additional non-tabular data.

3.2 EVALUATING TEXT EXTRACTION METHODS

In the related work chapter (see Chapter 2), this thesis exclusively explored Kamlehs' [21] approach to text extraction. This decision was influenced by the fact, that Kamleh had already conducted rigorous testing on three of the most prominent python packages designated to this task, performed on the same type of scientific papers that are relevant to this thesis.

Among the findings, Kamlehs' [21] research identified pdfminer as the most effective tool for text extraction, notwithstanding its limitations in accurately reconstructing tabular data.

This thesis initially assumed that the capabilities of pdfminer, supplemented by an additional tool for table extraction, would suffice. However, as discussed in the previous section, the performance of these supplementary tools fell short of expectations, prompting a reevaluation. This led to the realization that a more advanced text extraction tool is required – one that is not only proficient in standard text extraction but also facilitates the extraction and reconstruction of tabular data.

The challenges posed by pdfminer in accurately extracting tabular data are underscored by an example involving a table from the paper by Akter et al. [3], which is depicted in Figure 3.1. Subsequently, Figure 3.2 demonstrates the outcome of using pdfminer to extract this table. It becomes clear that, although pdfminer manages to retrieve all table content, the sequence of table elements is often scrambled. This disorder significantly hinders the precise reconstruction of the table in its original configuration, highlighting the limitations of pdfminer in processing complex tabular information.

Given Kamlehs' [21] investigation into Python packages for PDF text extraction – which found them lacking in tabular data reconstruction capabilities – this thesis has expanded its investigation into alternative methods, most notably Optical Character Recognition (OCR) technology. While traditionally favored for converting handwritten or printed materials into digital form [20][28][19], it can also be used for text extraction from PDF documents, as demonstrated in the work by Damerow et al. [9], thereby showcasing its versatility.

From the available OCR tools, Tesseract[1] stands out as a prominent OCR engine. Although its pytesseract[2] Python package is primarily designed for image-based text extraction, a workaround involves first converting PDF files into images using pdf2image[3]. This technique was applied to the table from the paper by Akter et al. [3], shown in Figure 3.1 and the results show that when extracting the table using pytesseract the original structure is well-preserved and a reconstruction is feasible, as evidenced in Figure 3.3.

[1] https://github.com/tesseract-ocr/tesseract
[2] https://pypi.org/project/pytesseract/
[3] https://pypi.org/project/pytesseract/

While OCR, particularly pytesseract, preserves table structures well, it's not widely used for PDF data extraction for several reasons. Unlike tools like Tabula or Camelot, pytesseract doesn't output extracted tables in a CSV file, requiring additional steps to define column and row boundaries if a DataFrame-like format is wanted. Moreover, compared to text extraction tools like pdfminer, which exhibit minimal extraction errors, pytesseract tends to introduce typographical errors, potentially leading to inaccuracies in extracted values and text. Consequently, tables extracted via pytesseract necessitate further processing to ensure the reliability of the data extracted.

Table 5: Descriptive Statistics and Wilcoxon Rank-Sum Tests of Pre and Post-Study Responses to the Constructs

Constructs		Pre-Study				Post-Study					
	α	*M1*	*SD1*	*S1*	*K1*	*M2*	*SD2*	*S2*	*K2*	*V-val*	*p-val*
Community Oversight Model:											
Transparency	0.88	4.07	0.80	-0.74	0.70	4.36	0.67	-0.74	-0.49	1054*	**0.010**
Awareness	0.82	3.95	0.79	-0.76	0.84	4.37	0.68	-0.72	-0.45	1110.5***	**<0.001**
Trust	0.90	3.61	0.80	-0.29	0.11	4.28	0.85	-0.81	0.11	633.5***	**<0.001**
Individual Participation	0.87	3.78	0.83	-0.67	0.59	4.23	0.73	-0.89	0.17	897.5***	**<0.001**
Community Participation	0.88	3.86	0.80	-0.56	0.24	4.18	0.83	-0.92	0.23	1002**	**0.002**
Community Trust	0.87	4.03	0.87	-0.85	0.45	4.22	0.71	-0.78	-0.03	1412*	**0.048**
Community Belonging	0.91	4.09	0.67	-0.53	-0.52	4.20	0.68	-0.60	-0.88	1899	0.209
Community Collective Efficacy	0.91	3.80	0.78	-1.09	2.56	4.12	0.68	-0.44	-0.36	1287***	**<0.001**
Self Efficacy	0.85	3.95	0.64	-0.58	0.79	4.32	0.61	-0.80	0.27	1021***	**<0.001**

*p<.05; **p<.01; ***p <.001

Figure 3.1: Example of a table as illustrated in Akter et al. [3].

```
↟Table 5: Descriptive Statistics [...]   [...]        [...]    [...]    [...]

Constructs                               Pre-Study    0.80     4.36     -0.49
                                         M1           0.79     4.37     -0.45
Community Oversight Model:                            0.80     4.28     0.11
Transparency                             SD1          0.83     4.23     0.17
Awareness                                             0.80     4.18     0.23
Trust                                    S1           0.87     4.22     -0.03
Individual Participation                              0.67     4.20     -0.88
Community Participation                  K1           0.78     4.12     -0.36
Community Trust                                       0.64     4.32     0.27
Community Belonging                      Post-Study
Community Collective Efficacy            SD2          -0.74    0.67     1054*
Self Efficacy                            M2           -0.76    0.68     1110.5***
                                                      -0.29    0.85     633.5***
*p<.05; **p<.01; ***p <.001              S2           -0.67    0.73     897.5***
                                                      -0.56    0.83     1002**
α                                        K2           -0.85    0.71     1412*
                                                      -0.53    0.68     1899
0.88                                     V-val        -1.09    0.68     1287***
0.82                                                  -0.58    0.61     1021***
0.90                                     p-val
0.87                                                  0.70     -0.74    0.010
0.88                                     4.07         0.84     -0.72    <0.001
0.87                                     3.95         0.11     -0.81    <0.001
0.91                                     3.61         0.59     -0.89    <0.001
0.91                                     3.78         0.24     -0.92    0.002
0.85                                     3.86         0.45     -0.78    0.048
                                         4.03         -0.52    -0.60    0.209
[...]                                    4.09         2.56     -0.44    <0.001
                                         3.80         0.79     -0.80    <0.001
                                         3.95
                                                      [...]    [...]
                                         [...]
```

Figure 3.2: Result of using PDFMiner to extract data from the table shown in Figure 3.1. The extracted text has been strategically segmented and rearranged into five adjacent sections to optimize horizontal space utilization and facilitate ease of reading.

```
Table 5: Descriptive Statistics and Wilcoxon Rank-Sum Tests of Pre and Post-Study Responses to the Constructs

Constructs Pre-Study Post-Study

a MI SDI SI KI M2 SD2 S2 K2 V-val p-val
Community Oversight Model:
Transparency 0.88 4.07 0.80 -0.74 0.70 4.36 0.67 -0.74 -049 1054* 0.010
Awareness 0.82 3.95 0.79 -0.76 0.84 437 0.68 -0.72 -045 1110.5*** <0.001
Trust 0.90 3.61 080 -0.29 O11 428 085 -081 O11  633.5*** <0.001
Individual Participation 0.87 3.78 0.83 -067 0.59 4.23 0.73 -0.89 0.17  897.5*** <0.001
Community Participation 0.88 3.86 0.80 -056 0.24 4.18 0.83 -0.92 0.23 1002** 0.002
Community Trust 0.87 4.03 0.87 -085 045 422 071 -0.78 -0.03  1412* 0.048
Community Belonging 0.91 4.09 0.67 -053 -052 420 0.68 -0.60 -0.88 1899 0.209
Community Collective Efficacy 0.91 3.80 0.78 -1.09 256 4.12 0.68 -044 -0.36  1287*** <0.001
Self Efficacy 0.85 3.95 0.64 -058 0.79 432 0.61 -080 0.27 1021*** <0.001

*p<.05; **p<.01; ***p <.001
```

Figure 3.3: Result of using pytesseract for data extraction from the table in Figure 3.1, with characters misinterpreted or missed by pytesseract highlighted in red.

Despite its limitations, this thesis identified pytesseract as the sole text extraction tool capable of preserving the structural integrity of tabular data during content extraction, deeming it worthy of further testing. Unlike Tabula or Camelot, which extract tables independently, pytesseract retrieves tables as part of the overall text extraction from a document. Consequently, comparing the quantity of tables extracted by pytesseract against these other tools directly is impractical. Instead, this analysis categorizes tables based on their structural preservation post-extraction into 'well contained', 'partially contained', and 'not contained'. 'Well contained' refers to tables extracted as a single text block, with allowances for separate table headers. 'Partially contained' describes tables whose majority remains intact in one text block, with minor segments isolated into one or more additional blocks. Lastly, 'not contained' tables are those fragmented into several smaller text blocks, complicating the identification of their original layout.

Applying this classification to the tables from the papers used to evaluate Tabula and Camelot, this paper achieves the results in Table 3.2. Calculating pytesseract's F_1-Score yields 0.71, despite classifying 'partially contained' tables as false negatives – a stricter criterion than used for Tabula and Camelot. Notably, the F_1-Score rises to 0.88 for quantitative tables, underscoring pytesseract's utility for tables with numeric values, which are more relevant to this thesis's objectives, as statistical values are predominantly found within these. This distinction was not applied to Tabula and Camelot, as it didn't markedly affect their scores. However, it's important to note that this evaluation doesn't account for false positives in pytesseract's evaluation due to its preliminary stage. Despite this, pytesseract stands out as the most promising tool for extracting tables from scientific papers relevant to this thesis, as it demonstrates potential for superior extraction rates compared to Tabula or Camelot.

This preliminary evaluation underscores the decision to pursue a pytesseract-based table extraction method in this thesis. Anticipating to achieve the initially calculated F_1-Scores for pytesseract, the development and subsequent broader-scale testing of this method are seen as crucial steps forward. This approach aims to leverage pytesseract's potential to significantly improve table extraction from scientific papers, highlighting its promising capabilities in comparison to existing tools.

Table 3.2: Table classifications using pytesseract

Table Classes	pytesseract	Quantitative	Qualitative
Well contained	21	19	2
Partially contained	5	3	2
Not contained	12	2	10
Total Number of Tables Present	38	24	14

3.3 Development of OCR-based table extraction

The forthcoming phase of this thesis involves pinpointing and segregating tables from text extracted via pytesseract. This step is crucial for independently processing textual and tabular data, enabling tailored analysis strategies for each data type.

The first idea that was tried to achieve this, involved the use of NLP. The idea was to use binary classification on each extracted text segment (i.e. segments separated by two or more newlines) and classify the segments into either belonging to a table or a paragraph. A big positive of this approach would be that consecutive text segments, classified as belonging to a table, could be stitched together, so that even tables classified as 'not contained' could be reconstructed. But this thesis encountered several challenges with the binary classification approach. Firstly, no existing database was found that classified text segments as either belonging to a table or a paragraph, nor was any similar database discovered. This wouldn't be a significant issue if the needed training data remained within a manageable scope for this thesis while still yielding satisfactory results. However, initial testing with a pre-trained Zero-Shot Classifier[4] revealed that while larger text segments were usually correctly classified, smaller segments from larger tables were often misclassified as being part of a paragraph.

The potential benefit of concatenating small text segments from tables was offset by the lack of specialized training data for binary classification. Additionally Table 3.2 indicates only two quantitative tables were segmented into multiple text blocks across five papers, suggesting the need for a vast number of papers to compile an adequate dataset. Efforts to develop and test binary classification using a pre-trained Text Classification model[5], on a small dataset compiled from 2023 SOUPS papers yielded only slight improvements over the Zero-Shot Classifier. Given the extensive resources and time required to curate the necessary dataset, this approach was deemed impractical within the confines of this bachelor thesis, despite recognizing its potential. The next strategy explored for differentiating tabular from textual data was a RegEx-based approach. Although initially dismissing RegEx for statistical value extraction due to the diverse types and presentations of these values, it proved highly effective for identifying table captions. An analysis of all 2023 SOUPS papers and selected articles from other journals showed that two RegEx patterns successfully located 100% of table captions, all whilst giving zero false positive hits. These patterns, targeting 'Table X:' or 'Table A.X:' (RegEx form shown in Listing 3.1) where X represents any number sequence, typically mark the beginning of a table caption, a reliable indicator of nearby tabular data.

[4]https://huggingface.co/facebook/bart-large-mnli
[5]https://huggingface.co/distilbert/distilbert-base-uncased-finetuned-sst-2-english

Listing 3.1: Python RegEx to locate table captions.

```
1 PDF_TABLE_PATTERNS = (r'\nTable\s\d+:', r'\nTable\s\w\.\d+:')
```

Having discovered a dependable method for locating table captions in text extracted by pytesseract, it's crucial to analyze how tables manifest in this extracted text. This step will enable effective extraction of tabular content, regardless of its presentation. To facilitate this, text from all 2023 SOUPS papers was extracted using pytesseract, allowing for a comprehensive examination of how tables are represented in the extracted format. The following key observations were made:

1. **Location of Tabular Data Relative to Its Caption:**
   Most tabular data in the analyzed papers is positioned after its table caption, with exceptions where data precedes the caption. A rare occurrence is both arrangements in the same paper, such as seen in Hazazi et al. [15], depicted in Figure 3.4. Thus, an evaluation is essential for each extracted caption to determine the tabular content's position relative to it.
2. **Gap Between Tabular Data and Its Caption:**
   Typically, pytesseract extracts tables into two text blocks: one for data and another for its caption. However, variations exist, including instances where tabular data and its caption form a single text block, as demonstrated in Figure 3.5. This necessitates verifying whether tabular data is already included within the caption block for every detected table caption.
3. **Gap Between Table Headers and Content:**
   While usually combined in one block of text, it is not uncommon for table headers to be separate from table data, a phenomenon also observable in Figure 3.4. Therefore, extracting a table requires inspecting for a small text block (up to two lines) above the data to include it, ensuring headers and data are correctly amalgamated.
4. **Internal Gaps Within Tabular Contents:**
   As mentioned, tabular data can be segmented into multiple blocks of text, previously categorized as 'partially contained' and 'not contained' tables. 'Partially contained' tables typically have minor segments detached. This behavior, possibly due to font changes or indentation, splits the data, as illustrated in Figure 3.6. During extraction, it's vital to assess whether any smaller adjacent text blocks are part of a table to prevent data loss. Tables identified as 'not contained' frequently disperse into many small text blocks. While this segmentation primarily affects to this thesis irrelevant qualitative tables, it becomes problematic on the rare occasions it involves quantitative tables. Such a segmentation demonstrated in Figure 3.7, suggests that consecutive small text blocks near a table caption may need consolidation to reconstruct the original table accurately.
5. **Missing Whitespace in Extracted Content:**
   A significant challenge in text extraction using pytesseract lies in its handling of spaces within and between texts. Notably, pytesseract limits the use of consecutive newlines to two, meaning that regardless of the actual spacing in the document, paragraphs will appear uniformly spaced in the extracted text. This uniformity can be beneficial by reducing the variance in gaps between text blocks that needs to be accounted for.
   However, pytesseract also constrains the use of spaces and similar whitespace characters to a single instance between elements. Consequently, the spacing between elements in a table is always represented by a single space character. This limitation can lead to structural loss,

particularly when table entries are empty and not explicitly marked with 'N/A'. This issue is illustrated in Figure 3.8.
Therefore, when extracting values from a table, it is crucial to ensure these values are correctly sorted into their respective columns. While achieving accurate placement may not always be feasible, the values in different columns often belong to distinct numerical ranges, which can provide clues to their original arrangement.

Armed with insights into the presentation of tables in text extracted by pytesseract, the next essential phase involves finally developing a method capable of extracting tabular data from its surrounding text, so both can be analyzed independently from each other.

```
[...]
Reason for turning on notifications Count

Get alerts when the deadbolt is jammed 10
Get alerts about who is accessing the house 8
Get security alerts 4
Get battery alerts 1

Table 1: Reasons for enabling smart lock notifications.
[...]
```

```
[...]
Table A.1: Study participants demographic information

Participant Gender Age group Education Time spent using the smart lock Connection to the internet
Pl Female 18-25 Bachelor's More than 4 months Directly (has a built in Wi-Fi)
P2 Male 26-35 Graduate student More than 4 months Wi-Fi hub (bridge)
P3 Female 26-35 Bachelor's 2-4 months Wi-Fi hub (bridge)
[...]
```

Figure 3.4: Table captions, highlighted in red, and tabular data, marked in blue, extracted from Hazazi et al.'s paper [15] using pytesseract. This illustration demonstrates the variability in positioning of table captions relative to the tabular data.

```
[...]
Exposure to AI A Moderate Amount A Great Amount 0.74 | 0.000
Exposure to AI A Moderate Amount A Little Bit 0.86 | 0.025
Exposure to AI A Moderate Amount A Lot 0.88 | 0.039
Table 8: Odds of a respondent sharing a theme coded as Highly Personal in their [...]
AI. Reporting includes all data, irrespective of p < 0.05.
[...]
```

Figure 3.5: Table caption, highlighted in red, and tabular data, marked in blue, extracted from Kelley et al.'s paper [22] using pytesseract. This illustration demonstrates the inconsistency in distance between table captions and tabular data.

Table 6: Logistic regression on the correctness of evaluations, as dependent on the used process and the alert category

Variable	Coeff.	OR change (%)	*p*-value
(Intercept)	2.56	NA	<0.001
Process	**0.98**	**167.0**	**0.035**
Reference category: Scan			
Category : CnC	-1.20	-69.8	0.070
Category : Malware	**-1.89**	**-84.9**	**0.002**
Category : Policy	-0.76	-53.1	0.406

```
[...]
Table 6: Logistic regression on the correctness of evaluations,
as dependent on the used process and the alert category

Variable Coeff. OR change (%) p-value
(Intercept) 2.56 NA <0.001
Process 0.98 167.0 0.035
Reference category: Scan

Category : CnC -1.20 -69.8 0.070
Category : Malware -1.89 -84.9 0.002
Category : Policy -0.76 -53.1 0.406
[...]
```

Figure 3.6: Original and pytesseract-extracted versions of a table from Kersten et al.'s paper [23], with tabular data highlighted in blue and purple. This figure illustrates the impact of font variations and indentation on the textual representation by pytesseract.

```
[...]
Table 2: Computer and email expertise demographics of survey participants

Support Group RandomGroup Control Group

N (%) N (%) N (%)

Computer Familiarity

Work in or hold a degree in CS/IT 12 (20%) 6 (10%) 8 (13%)

Do not work in or hold a degree in CS/IT 48 (80%) 54 (90%) 52 (87%)
Computer Expertise
[...]
```

Figure 3.7: Table caption, highlighted in red, and tabular data, marked in blue, extracted from Liu et al.'s paper [25] using pytesseract. This illustration demonstrates how some tables have their data extracted in many small segments.

Table 8: Correlation between SSBS Technical (**T**) and Social (**S**) Scales

	T1	T2	T3	T4	T5	T6	T7	T8	S1	S2	S3	S4	S5	S6
T1	1.000	.455	.594	.401	.327	.339	.460	.270	-.196	-.108	-.159	-.047	-.065	-.002
T2		1.000	.486	.405	.342	.286	.401	.344	.009	.024	.007	.118	.083	.112
T3			1.000	.484	.393	.432	.445	.282	-.017	.018	-.011	.106	.036	.102
T4				1.000	.302	.363	.282	.239	.004	.078	.062	.100	.141	.103
T5					1.000	.472	.352	.098	.045	.029	.001	.119	.066	.002
T6						1.000	.243	.169	.017	.043	.020	.120	.141	.029
T7							1.000	.199	-.033	.087	.033	.051	.037	.072
T8								1.000	.009	.053	.053	.083	.041	.122
S1									1.000	.612	.616	.473	.537	.420
S2										1.000	.629	.498	.515	.444
S3											1.000	.545	.484	.475
S4												1.000	.423	.401
S5													1.000	.381
S6														1.000

```
[...]
Table 8: Correlation between SSBS Technical (T) and Social (S) Scales

TI 12 T3 T4 TS T6 17 T8 SI $2 S3 S4 $5 S6
T1 =| 1.000 | 455 594 401 327 339 460 270 -.196 -.108 =.159 =.047 =.065 =.002
12 1.000 486 405 342 286 401 344 009 024 007 118 083 2
T3 1.000 484 393 432 445 282 -.017 018 -.011 106 036 102
14 1.000 302 363 282 239 004 078 062 100 141 103
TS 1.000 472 352 098 045 029 001 19 066 002
T6 1.000 243 169 017 043 020 120 141 029
17 1.000 199) =.033 087 033 051 037 072
T8 1.000 009 053 053 083 041 122
SI 1.000 612 616 473 537 420
S2 1.000 629 498 515 444
$3 1.000 545 484 475
S4 1.000 423 401
$5 1.000 381
S6 1.000
[...]
```

Figure 3.8: Original and pytesseract-extracted versions of a table from Huang et al.'s paper [17], with tabular data highlighted in blue and purple. This figure demonstrates that the presence of whitespace within tabular data can result in a loss of structural integrity when extracted.

3.4 Workflow of the table extraction method

This section outlines the development of the table extraction method, providing a comprehensive, step-by-step guide to its workflow. Each step's significance is explained in detail, including the rationale behind the exclusion of certain steps. This overview aims to offer clear insights into the methodological underpinnings and operational nuances of the proposed table extraction process.

The table extraction process begins once text from a scientific paper has been extracted using pytesseract. The subsequent steps aim to isolate tables from the extracted text:

1. **Identifying Table Captions with RegEx:**
   This step utilizes the aforementioned RegEx patterns, to search for mentions of 'Table X:' and 'Table A.X:'. These pinpoint the initial locations of table captions, facilitating the extraction process.
2. **Processing Each Identified Table Caption:**

   a) **Aggregating Content Above the Caption:**
   This step involves identifying the immediate text block preceding the caption, regardless of direct connection, using the caption's starting point as a divider. Additionally, checks ensure no data is accounted for twice across captions and that potential header information in adjacent single/double-line text blocks is included.

   b) **Aggregating Content Below the Caption:**
   This step aims to identify the immediate text block following the caption, however initial assessments need to check whether the text block containing the caption already includes subsequent data, judged by the text block's length. If this is the case, it is assumed that possible tabular data may already be integrated within the caption block. If not, the following text block is identified, while again accounting for possible single/double-line text blocks that might hold header information.

   c) **Assessing Numeric Token Ratios Above and Below the Caption:**
   By applying the following formula to the text content above and below the caption, it is possible to determine the numeric ratio of these contents:

$$\text{numeric ratio} = \frac{\text{numeric tokens}}{\text{total tokens}} \tag{3.1}$$

   This metric, where a 'token' is classified as a character sequence delimited by whitespace and accounted for as numeric if it contains at least one numeric character, helps in evaluating the likelihood of tabular data presence based on numeric value density. A higher numeric ratio suggests that the analyzed content is more likely to be tabular, especially in the case of quantitative tables, which tend to contain significantly more numeric tokens than typical text paragraphs.

   d) **Identifying Likely Tabular Data:**
   By comparing numeric ratios, the method determines each content's probability of containing tabular data, requiring a minimum of 20 tokens and a numeric ratio above 0.15 for consideration. These criteria are established to exclude primarily text-based qualitative tables, as they seldom contain the statistical information pertinent to this thesis's objectives and are hard to differentiate from paragraphs.
   When at least one of the analyzed contents meets the minimum requirements, the one with the higher numeric ratio that satisfies these criteria is flagged as containing tabular data related to the corresponding table caption.

   e) **Content Extraction:**
   Content flagged as containing tabular data, along with its caption, is extracted for further analysis and saved separately.

3. **Excluding Extracted Content from Further Text Analysis:**
   Once all table captions have been processed, all content that was flagged as containing tabular data is removed from the primary text corpus, to allow for distinct analysis paths for textual and tabular data.

This table extraction process is visualized in Figure 3.9.

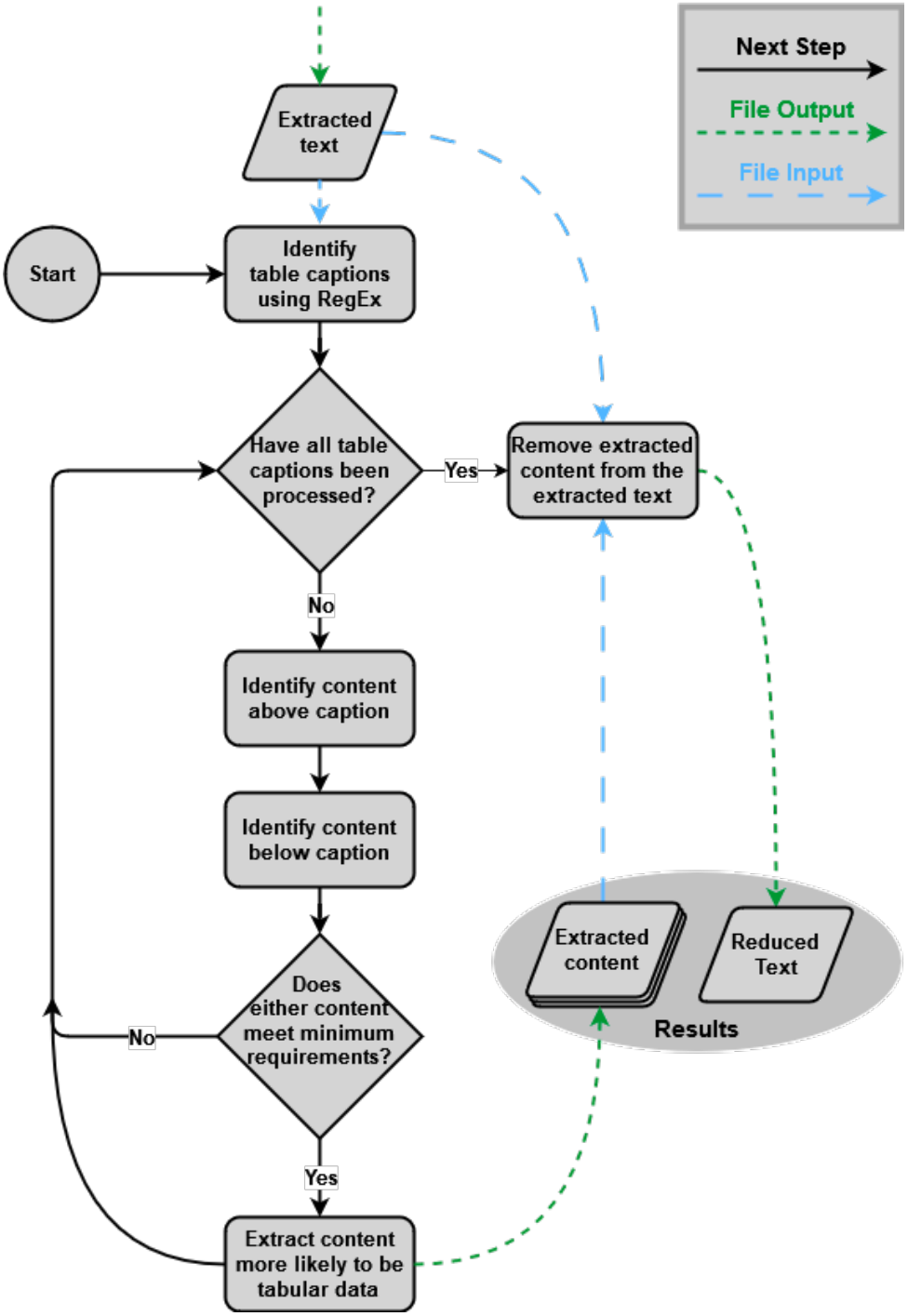

Figure 3.9: Flowchart Illustrating the Table Extraction Process: This figure outlines the sequential steps involved in extracting tables from scientific paper texts processed by pytesseract.

3.4.1 FEATURES NOT IMPLEMENTED

Earlier in this chapter, this thesis discussed extracting 'not contained' tables, noted for their segmented appearance. This feature, however, was not included in the final method due to the difficulty of distinguishing between fragmented tables and figures in text processed by pytesseract. Testing showed that figure content appears segmented similarly to 'not contained' tables (refer to Figure 3.10) and frequently contains enough numeric tokens to meet the minimum requirements as to not be filtered out. This leads to potential misclassification as tabular data. Given that figures often occur near table captions, this confusion resulted in the extraction of significant irrelevant data. Therefore, to maintain the extraction process's accuracy, this feature was excluded.

Another feature that was notably omitted from the final version of this method was the aggregation of 'partially contained' tables, which, unlike their 'not contained' counterparts, consist largely of content within a single text block with only minor data elements missing. The initial strategy to address this involved, upon identifying the first text block adjacent to a table caption, assessing whether the following text block met the established minimum numeric ratio criteria for table determination. If so, this block and any subsequent ones matching the criteria would be annexed to the table.

However, it became apparent that the effectiveness of the minimum numeric ratio as a predictive tool decreases with distance from the original table caption. While this ratio effectively filters out non-quantitative data close to the caption – due to a preference for content with a higher numeric ratio – it proved insufficient for distinguishing between table data and generic text further away. This inadequacy frequently resulted in the erroneous inclusion of non-table data within what were supposed to be 'fully contained' tables, thereby introducing more errors than it resolved.

Adjusting the minimum numeric ratio upwards could have mitigated this issue, but such an adjustment risked excluding legitimate table data, resulting in many 'partially contained' tables remaining as such. Ultimately, this thesis could not identify an optimal numeric ratio that balanced false positives and false negatives effectively. Consequently, rather than incorporating an imperfect solution which may have slightly improved performance, the decision was made to exclude this feature entirely, leaving the refinement of this aspect for future research.

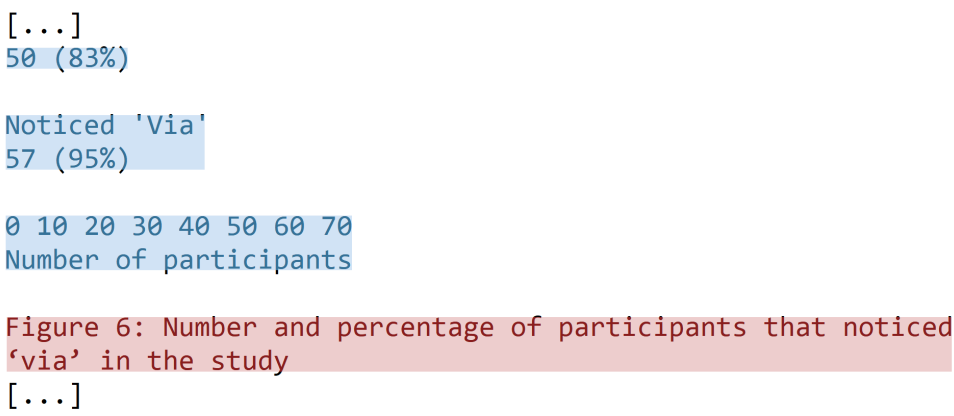

Figure 3.10: Figure caption, highlighted in red, and figure data, marked in blue, extracted from Liu et al.'s paper [25] using pytesseract. This illustration demonstrates how pytesseract extracted figures, share a similar segmentation to "not contained" tables (see Figure 3.7).

3.5 Extracting statistical values from pytesseract processed data

Having developed a method to separate tabular and textual data, the next phase of this thesis focuses on extracting pertinent statistical values from the identified data. The related work section outlines two methodologies for addressing this challenge, as discussed in Section 3.

Initially, RegEx were considered but swiftly deemed unsuitable due to constraints highlighted by Kamleh [21]. These limitations are now even more pronounced, given that the chosen text extraction tool for this thesis frequently introduces typographical errors, undermining the reliability of a RegEx-based method.

The approach used by Pires [34] seemed more fitting to this thesis's method. He deployed a LLM, specifically OpenAI's Assistant API to perform statistical tests and analysis on tabular data. This approach appeared to be more viable, as LLMs offer the advantage of contextual understanding and a higher tolerance for typographical errors. Given these considerations, a LLM approach was the first to be explored in addressing the challenge of extracting statistical values from text and tables.
While the task undertaken by Pires [34] – extracting statistical data from tables, as well as conducting tests – may seem more complex, he benefited from working with data in a CSV format, ensuring a high degree of reliability and correctness. In contrast, the aim of this thesis, though focused solely on data extraction, confronts the additional challenge of data integrity. The table data in this thesis, is provided in a TXT format, cannot guarantee accuracy regarding typography, structure, and content, nor can it definitively confirm its classification as tabular data, making the task uniquely demanding despite its narrower scope. However a LLM approach may be able to help in many of these aspects.
Initially, the expectation was that the Assistant API could assist in converting tabular data from its TXT format to a CSV format, and it often succeeded in this with minimal errors. However, it soon became apparent that this conversion step could be integrated directly into the extraction process. Since the focus of this thesis was not on the table in its CSV format but rather on the statistical values it contained, the direct integration offered a more streamlined approach.

The extraction phase introduced the initial challenges. This thesis initially aimed to tally the total numbers of statistical values in the text, checking if the totals matched expectations (e.g. equal number of p-values and effect sizes). However, when using the Assistant API to count these statistical values, discrepancies quickly emerged. The counts were often inaccurate, particularly with longer tables or if multiple tables at once were analyzed, and varied significantly between requests – even with identical prompts. Sometimes the count was too high, sometimes too low, but the Assistant API rarely provided an accurate count. This inconsistency has been noted by others, as seen in discussions about OpenAI's LLMs' difficulties with accurate counting[6,7].

In response to these challenges, a new strategy was attempted: outputting the specific values and providing a count for each, hoping the more granular approach would improve accuracy. This method did indeed yield better results, though the counts were still occasionally off. The final and most successful approach eliminated counting altogether, focusing on outputting each value identified as a statistical value of interest. This strategy proved to be the most reliable, offering consistency across separate requests and generally achieving high accuracy. This approach could be easily expanded on by counting the number of values provided in the output, but ultimately this feature was deemed unnecessary and was left out.

[6] https://community.openai.com/t/chatgpt-cannot-count-words-or-produce-word-count-limited-text/47380
[7] https://genai.stackexchange.com/questions/43/does-chatgpt-know-how-to-count

3.5.1 FORMATTING THE DATA FOR EXTRACTION

With a functioning basic prompt for data extraction in place, the next consideration was how to present the extracted data to the Assistant API for analysis. The data could be processed table by table or all at once. Analyzing data table by table typically reduces errors due to smaller amounts of content being processed per request but requires more requests, increasing runtime and API costs. It was determined that analyzing all tables at once, while slightly increasing errors, significantly reduced both runtime and costs, which is why this approach was chosen. If accuracy is paramount to future work aiming to integrate this method, it can easily be adjusted to analyze data in a more modular fashion.

After extraction from the paper, the tabular data, along with their captions, are recorded in a TXT file, with each table separated by hyphens. This format allows all the tables to be submitted to the Assistant API for statistical analysis at once, with each one clearly separated.

While handling the remaining text seemed straightforward, challenges arose when the text was analyzed. Although feasible, the extraction accuracy was suboptimal, suggesting a need for a more structured approach. Utilizing the paper's outline to divide the text into chapters allows for individual analysis of each section, substantially reducing the bulk of text to be processed simultaneously. To further cut API costs, non-essential sections can be omitted. Such a segmentation is likely not necessary in the near future, as the models the Assistant API relies on are continuously being improved upon by OpenAI. For this thesis, only the chapter containing research results was analyzed to maximize cost-efficiency, as these typically report the majority of statistical values.

Since not all results sections are explicitly labeled as such, an additional step was implemented to isolate just the results section for analysis. For this, all chapter titles were submitted to the Assistant API, which then identified the chapter most likely to contain research results. RegEx were used to locate the start and end of this chapter in the text, thus isolating its contents for focused extraction. This method ensures that only the relevant section is analyzed, enhancing both efficiency and cost-effectiveness.

It was also observed that inputting data directly into the prompt in string format tended to enhance accuracy, especially given the occasional issue where the Assistant API would mark a session as 'completed' without delivering an actual response when data was provided as a file – an issue that has been documented in several community posts[8,9]. Consequently, in the final version of this thesis, the table data is supplied to the Assistant API in string format. However, this method is not used for the text from the chapter containing research results, as the length of these chapters sometimes surpasses the message token limit, currently set at $32,768$ tokens [31]. For files, the Assistant API accommodates a much higher token limit of $2,000,000$, making file submission feasible for extensive texts [29]. Although the token limit is to be raised in the future, another potential solution to the non-response issue could involve segmenting the text of the results chapter into smaller parts before submission to the API, so that it fits into the prompt token limit. This approach was not pursued in this thesis, as the Assistant API remains in its beta phase, and it is anticipated that such errors will be resolved upon its full release.

3.5.2 ENGINEERING THE PROMPTS FOR VALUE EXTRACTION

The segmentation of tabular and text data allows this thesis to engineer tailored prompts, or instructions as they are referred to in the case of the Assistant API, for each type of data independently.

[8] https://community.openai.com/t/anyone-experiences-no-message-response-using-assistants-api/656254
[9] https://community.openai.com/t/can-assistant-complete-run-without-any-message-generated/533491/2

Prompt engineering encompasses various strategies and tactics, with the performance of each of these still not being fully assessed, given its nascent status as a field of research. OpenAI has provided a guide to prompt engineering [30], outlining approaches to achieve improved output results from their models. Drawing from this guide, the prompts for extracting statistical data from tables and text, as seen in Listings A.1 and A.2, were crafted. These prompts aim to encompass every relevant detail for performing the extraction task on said data and dictate the precise sequence of action in which it is to be performed. The goal is to ensure that the Assistant API tackles the task consistently across requests, thereby reducing variability in the quality of results.

Moreover, by providing a clear template for the expected output of the Assistant API, the results of requests are standardized, enabling further processing of the data.

As mentioned earlier, the text data intended for analysis by the Assistant API will be provided in the form of a file. However, files can also be passed to the Assistant API not only as part of a message but also during its creation. This capability allows additional information to be provided to the Assistant API outside of its prompt, which it can consult when handling requests. In this thesis, such a file was prepared for the Assistant API, containing a list of commonly used effect sizes. Effect sizes, unlike other statistical values like p-values and confidence intervals, manifest in various forms, potentially elongating the prompts significantly if all variations were listed. Instead, this thesis adopted a cleaner approach by creating a separate list and referencing it within the prompt. This way, the Assistant API can refer to its files to determine which types of effect sizes exist and look for corresponding values.

With the prompts successfully engineered, the last phase of the extraction method can be implemented, thereby completing the approach. The flowchart for the finished approach showing the full workflow from the original PDF file to the extracted statistical values is illustrated in Figure 3.11.

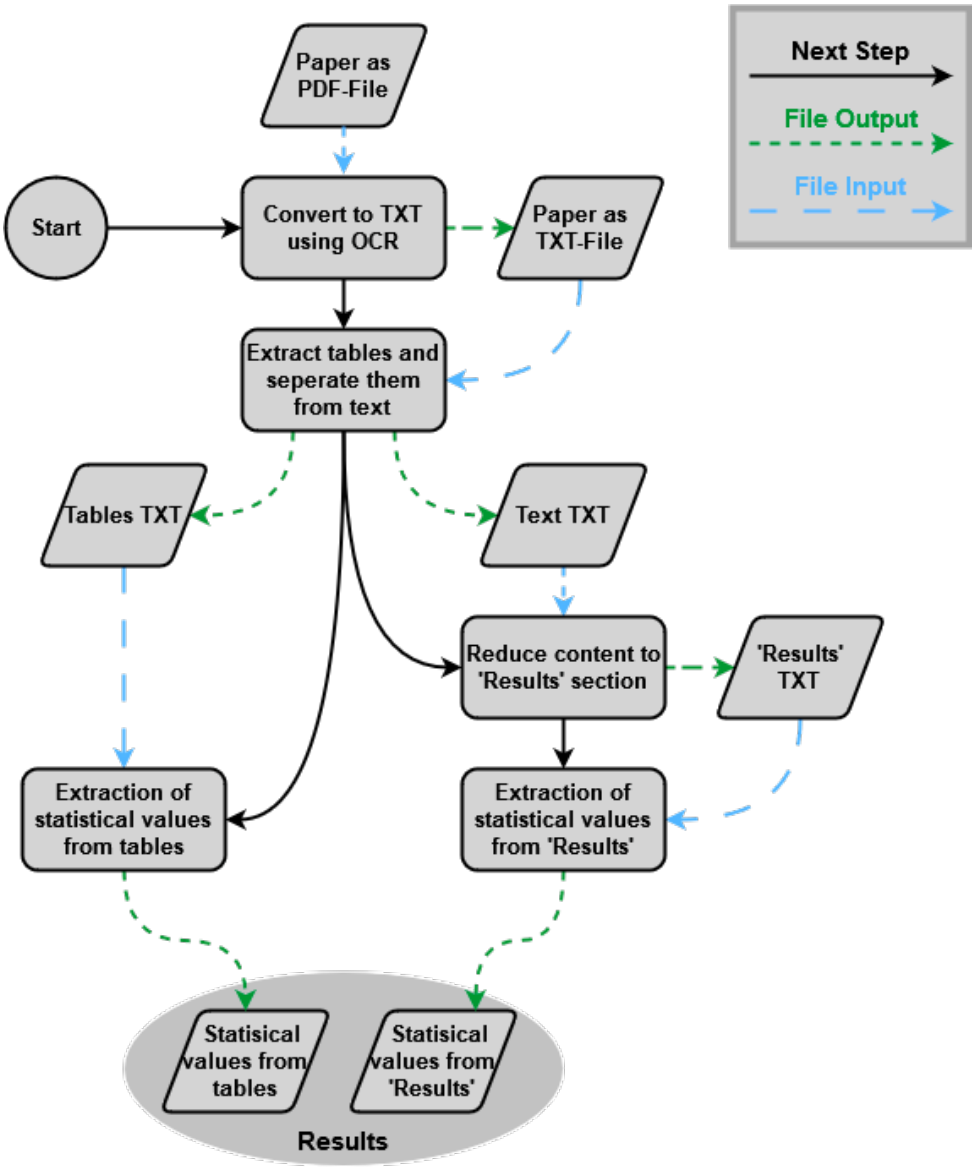

Figure 3.11: Flowchart Illustrating the Extraction Process of Statistical Values: This figure outlines the sequential steps involved in extracting statistical values from a scientific paper in PDF format.

4 TESTING PHASE

This chapter will discuss the testing phase of the thesis, focusing specifically on the test setup, execution and the produced results.

4.1 TEST SETUP

The initial step in the test setup involved selecting the papers for examination. It was decided to conduct the tests on 15 papers from the Symposium on Usable Privacy and Security (SOUPS) and 15 from the Conference on Human Factors in Computing Systems (CHI). These papers were sourced from the years 2020, 2021, and 2022, with five papers chosen from each year for each conference. The selection criterion was that each paper must include at least one table featuring p-values, effect sizes, or confidence intervals, and provide an outline. However, from both conferences, some papers were intentionally selected without any statistical values in their tables. This was done to assess whether the Assistant API could accurately identify the absence of these values in all tables or if it would fabricate a false answer.

The selection of the conferences from which the papers were sourced was deliberate and their field of HCI is relevant to this thesis' supervisors vision for further research. SOUPS was chosen because its papers had already been utilized for refining the developed method. CHI was selected because it is also a conference within the same field and not only offers its papers in PDF format but also in HTML. This availability facilitates the development and testing of a separate method that utilizes the HTML format, which provides more reliable means for extracting tabular and textual data. By developing this alternative approach, a direct comparison can be made with a method that uses machine-readable content from the outset, potentially highlighting any issues with the original approach.

For the test, the API requests to extract statistical values from tables and text were executed three times for each paper. Then, all unique extracted values were aggregated, where for each unique value, only the highest number of occurrences from a single request was recorded. This approach was used to counteract the high variability typically seen in responses from chatbots. While aware that this approach might increase the occurrence of false positives, it was observed that the Assistant API tended to underreport values rather than overreport. Therefore, an aggregate approach was favored over one that only considers the most frequently reported values.

4.2 ALTERNATIVE HTML APPROACH

Developing the approach using papers in HTML format was more straightforward than the PDF approach. There are established public libraries capable of easily handling the extraction tasks that the PDF approach found challenging. The pandas library[1] includes a function specifically for extracting tables from HTML files into a DataFrame format, which in testing did not once fail. Additionally, the beautifulsoup4 package[2] facilitates the easy location of content based on its class, enabling the straightforward extraction of section titles, table captions, and other relevant content. The availability of these dependable extraction tools has significantly reduced the necessity for creating bespoke functionality for the HTML approach. This means that the development effort primarily focused on crafting a framework to combine these tools for the specific purpose of this thesis.

[1] https://pypi.org/project/pandas/
[2] https://pypi.org/project/beautifulsoup4/

While tables are readily extracted in a DataFrame format, this thesis opted to convert them into a TXT format for the HTML approach, with table elements delimited by commas. This conversion facilitated the straightforward inclusion of table captions above their corresponding tabular data, as well as allowing for the easy presentation of multiple tables within a single file. This conversion was adopted to mirror the format used in the PDF-based method, thereby enhancing comparability. Moreover, for the extraction of values, this thesis did not devise a unique prompt tailored to the HTML-based method. Instead, it employed the same prompts designed for the PDF method to preserve consistency in the testing framework between the two approaches. This choice may have marginally influenced the performance of the HTML approach, considering that the DataFrame format more effectively captures table structures, and certain elements of the prompt were specifically devised for the PDF method.
The workflow of this HTML approach is illustrated in Figure 4.1.

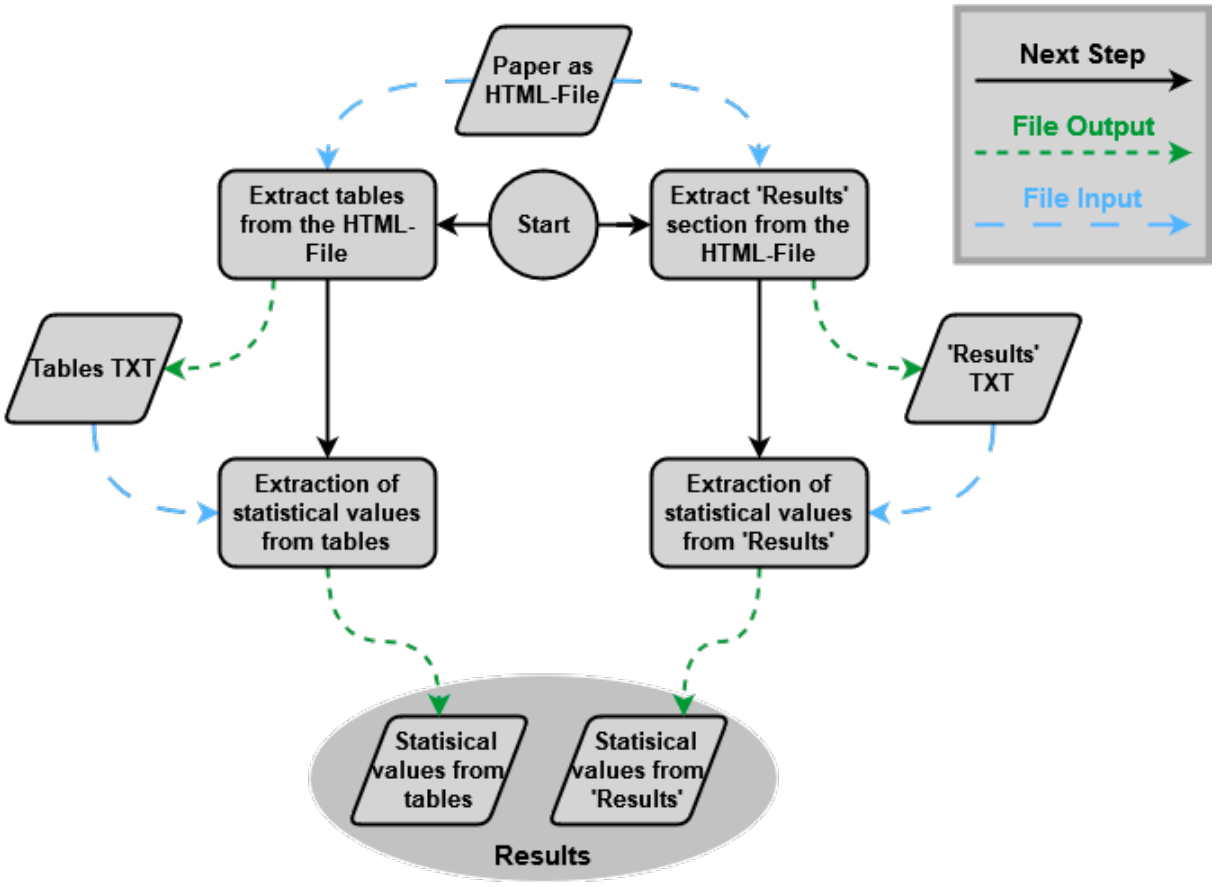

Figure 4.1: Flowchart Illustrating the Extraction Process of Statistical Values: This figure outlines the sequential steps involved in extracting statistical values from a scientific paper in HTML format.

4.3 Test results

The approaches were tested on several points of interest:

- Accuracy in extracting quantitative tables.
- Accuracy in extracting statistical values from tables.
- Accuracy in extracting statistical values from text.

The results from these tests are documented in Table 4.1. The F_1-scores reported in this table were calculated in a non-weighted manner, where each paper's individual F_1-scores contributed equally to the mean and median scores reported. This means that regardless of the number of numeric tables or statistical values contained within each paper, their impact on the overall mean and median scores was uniform.

The F_1-Scores for table extraction were calculated in a manner similar to that used for Tabula and Camelot, where both fully and partially extracted tables count as true positives. However, a key difference in this approach is that extracted qualitative tables are considered false positives, akin to the extraction of non-tabular data.

For the extraction of statistical values, typographical errors did not influence the classification of values as long as the original value could still be discerned from the text. However, values that were correctly extracted but originated from an inappropriate data structure, such as additional text appended to a table or from a residual table within the text data, were counted as false positives. This is because they are not part of the content originally intended for extraction in that phase of the process and would require further manual verification to confirm their accuracy.

Papers lacking statistical values in any of their tables were excluded from the calculation of the mean and median F_1-Scores for table value extraction. This exclusion was based on the absence of false positives in any of these papers. Including their F_1-Scores was considered relevant only if some of them had output false positives; otherwise, their inclusion could unduly skew the final F_1-Scores without reflecting actual performance in extracting statistical values.

Table 4.1: Mean and median F_1-Scores for extracting data from scientific papers, calculated from data recorded in the tables A.1, A.2 and A.3.

Paper Source and Format	Quantitative Tables		Table Values*		Text Values*	
	Mean	Median	Mean	Median	Mean	Median
SOUPS (PDF)	0.770	0.903	0.623	0.760	0.841	0.900
CHI (PDF)	0.659	0.800	0.582	0.630	0.827	0.860
CHI (HTML)	0.983	1.000	0.924	1.000	0.938	0.950

* P-Values, effect sizes and confidence intervals.

4.3.1 Interpreting the test results

There are several key takeaways from the results reported in Table 4.1. While the PDF-based approach yields similar F_1-Scores for extracting statistical values from text across both the analyzed SOUPS and CHI papers, with only minor performance differences, the disparity in F_1-Scores for the extraction of quantitative tables is more pronounced. Specifically, the SOUPS papers achieve significantly higher mean and median F_1-Scores. This discrepancy may stem from the fact that the tabular extraction method was primarily tested and developed using SOUPS papers as references. Consequently, tables from CHI papers, which likely feature unique

formats not addressed in Section 3.4, or have a higher incidence of 'partially contained' and 'not contained' tables, are less likely to be accurately extracted.

Another noticeable difference lies in the performance of extracting statistical values from text between the PDF and HTML approaches. Although this difference is also observed in table data extraction, it is more pronounced and expected there, given that extracting tabular data from HTML is significantly easier than from PDF, as previously discussed. One major factor contributing to this discrepancy is the inability of pytesseract to accurately recognize Greek characters, which are occasionally used to represent effect sizes. For instance, in the paper by Story et al. [36], some effect sizes are denoted by 'ε^2'. However, when extracted using pytesseract, these were misinterpreted as '$e?$' or '$\in?$', leading to their misidentification by the Assistant API and subsequent non-extraction. This issue was acknowledged prior to testing. Attempts to improve the extraction of ancient Greek letters using the specialized training data provided by Tesseract[3] yielded disappointing results. Less than half of the Greek letters were correctly extracted, and occasionally, Roman letters were erroneously recognized as Greek letters. Given the overwhelming number of Roman to Greek letters, even infrequent misinterpretations resulted in a significantly higher rate of false positives than true positives. This led to the decision to exclude this approach from testing. Another contributing factor to the differing performances between the HTML and PDF approaches in value extraction from text, is due to the extraction rate of quantitative tables. Unlike the HTML method, which consistently extracted all quantitative tables in their entirety, the PDF approach occasionally missed parts or entire tables containing the statistical values of interest. If such a table falls within the section containing research results, its values might erroneously be extracted during the wrong phase, subsequently being counted as false positives. To address this, the prompt for statistical values extraction from text explicitly states: "Exclude any values that appear to be part of tables or other structured elements rather than in the narrative text." Despite this precaution, these values are sometimes still mistakenly extracted. For instance, during tests on the paper by Jahanbakhsh et al. [18], their numeric table was not extracted, resulting in the values being incorrectly extracted during the phase of statistical values extraction from text, as documented in Table A.2. These false positives adversely impact the F_1-Score of the PDF approach, further elucidating why the HTML-based method demonstrated superior performance in extracting statistical values from text.

Another finding of this thesis was that while the Assistant API mostly struggled with extracting effect sizes from tables and text – due to typographical errors, misrepresentation of Greek letters, and the diversity of effect size representations – a particular type of table substantially impacted the p-value extraction rate. This table type does not directly list p-values; instead, it displays corresponding effect sizes and denotes p-value significance levels with asterisks next to them (see Figure 4.2). These asterisks range from 1 to 3, indicating significance levels of $p < 0.05$, $p < 0.01$, and $p < 0.001$, respectively. This setup frequently led to difficulties; although significance levels were typically explained in the table captions, this was not universally the case. Additionally, smaller characters such as asterisks are more prone to misrepresentation or omission by pytesseract. Furthermore, even when these explanations were present, the Assistant API sometimes struggled to interpret them correctly. A potential solution to this issue would be to incorporate these explanations and provide additional context directly within the prompt for extraction from tabular data, thereby reducing dependence on table captions. This approach was not implemented prior to testing because, although this thesis encountered a scenario where asterisks were used to denote significance levels, the asterisks were supplementary to the p-values, which allowed for the correct extraction of p-values from the table.

[3] https://github.com/tesseract-ocr/tessdata/blob/main/grc.traineddata

Table 2: Multi-level regression results on the effect of income and education on information practices during COVID-19. (Significance: $*p < 0.05$, $**p < 0.01$, $***p < 0.001$)

Dependent Variables		Middle income (vs. Low income) β (Std. Error)	Bachelor and higher (vs. Less than bachelor) β (Std. Error)
Platforms utilized to get COVID-19 info	Internet	0.054 (0.027)*	0.012 (0.020)
	TV	-0.004 (0.026)	-0.057 (0.019)**
	Radio	-0.027 (0.018)	-0.038 (0.013)**
	Newspaper	0.014 (0.003)	-0.034 (0.010)***
	Website	0.126 (0.024)***	0.094 (0.018)***
	Social media	0.081 (0.024)***	-0.056 (0.018)***
Specific social media platform	Facebook	-0.052 (0.026)	0.024 (0.019)
	Twitter	-0.001 (0.023)	0.063 (0.017)***
	Reddit	-0.026 (0.013)	-0.002 (0.010)
	YouTube	-0.116 (0.025)***	-0.050 (0.018)**

Figure 4.2: Excerpt of a table by Zhang et al. [47]. This is an example of a table using asterisks to denote significance level.

5 DISCUSSION

With the tests completed and their results interpreted, this section now transitions to discussing the relevance of the methods developed in this thesis for future research.
Although the OCR-based table extraction approach developed in this thesis does not yet yield optimal results, it represents a significant improvement over the best open-source alternatives for extracting tabular data from two-column PDF documents. The mean F_1-Score from the tested CHI papers was relatively low at $F_1 = 0.659$, yet this still surpasses the scores obtained using Camelot ($F_1 = 0.41$) and Tabula ($F_1 = 0.44$). However, it falls significantly short of the initially predicted F_1-Score, which was anticipated to be around 0.88 – a target only met by the median F_1-Score from the SOUPS tests. Additionally, the tables extracted by this method are not yet formatted into a DataFrame and exhibit issues related to typographical errors. These challenges could potentially be overcome by transitioning to a DataFrame format using an LLM, and by addressing typographical errors through image preprocessing.

This thesis acknowledges the potential of the OCR-based table extraction method developed here, as it surpasses commonly used alternatives in effectiveness. However, it also recognizes that further refinement is necessary.
When it comes to the extraction of statistical values, the methods developed in this thesis using the Assistant API show real promise. Although the F_1-Score for extraction from tables is low, with the lowest mean F_1-Score being $F_1 = 0.582$ from tests conducted on the CHI papers in PDF format, it is primarily due to the low rate of table extraction itself. If we consider only CHI papers where tables containing relevant values were fully extracted, we obtain a mean F_1-Score of $F_1 = 0.878$, indicating significant potential if applied separately from the table extraction process. For text extraction, the lowest mean F_1-Score is $F_1 = 0.827$. While these indicate a worse extraction rate than the one achieved by Kamleh [21] with her extraction rate of 89.8%, (only for value extraction from text), she also did extract entire sentences and failed to isolate the values of interest within. Additionally, this thesis's F_1-Scores were extraction results on tables lacking proper structuring and text prone to typographical errors. In contrast, the F_1-Scores from the HTML approach, with mean F_1-Scores of $F_1 = 0.924$ for value extraction from tables and $F_1 = 0.938$ for value extraction from text, highlight the potential improvements if such issues are addressed. Therefore, the PDF approach could be further optimized by, for example, continuing to use pytesseract for table extraction tasks, while adopting PDFMiner for text extraction tasks to minimize typographical errors, which would most likely lead to a clear improvement over Kamleh's approach.

Contrary to the table extraction method, the value extraction techniques in the PDF approach consistently produce robust results. Although there is room for further refinement, this thesis contends that these methods are sufficiently developed to be valuable tools for similar research in the future.
The PDF approach, while viable with future refinements, does not perform as well as the HTML method. The HTML approach achieves significantly higher F_1-Scores and avoids common problems such as loss of table structure and typographical errors. The main area for potential improvement in the HTML method concerns the prompts used for value extraction, which was the only aspect that encountered issues during testing. Consequently, despite the advancements this thesis has made in extracting values from two-column PDF files, it is advisable to use the HTML extraction method whenever possible, due to its superior performance and reliability.

6 Conclusion

The primary goal of this thesis was to develop a method that enables the extraction of statistical values from research papers in the PDF format, building upon the research conducted by Kamleh [21], who utilized a RegEx-based approach. While her approach successfully extracted 89.8% of sentences containing statistical parameters, it notably extracted entire sentences rather than isolating the parameters themselves. Her method's reliance on RegEx also limited its ability to recognize previously unseen notations of these parameters. Moreover, while effective in extracting from text, Kamleh's method could not extract statistical values from tables due to the limitations of PDFMiner, which presented tabular data in a format incompatible with a RegEx-based extraction.

To address these shortcomings, the LLM-based approach developed in this thesis aims to allow for value extraction based on context rather than mere syntax. Additionally, pytesseract was favored over PDFMiner for text extraction due to its superior ability to retain table structures. When tested, this approach achieved a minimum mean F_1-Score of 0.659 for table extraction, surpassing the best open-source alternatives tested on similar papers. The minimum F_1-Score for value extraction from tables stood at 0.582, but improved to 0.878 when considering only successfully extracted tables. For value extraction from text, the minimum F_1-Score was 0.827.

Although this thesis successfully integrated a method to extract tables and their values – unlike Kamleh's [21] RegEx-based strategy – it still requires significant refinement to be considered reliable for future use. While Kamleh's extraction rate from text may exceed that of this thesis, it is important to note that her method did not focus on explicitly extracting values but rather the sentences surrounding them. Furthermore, the superior results demonstrated by the HTML approach, which achieved a mean F_1-Score of 0.938 in value extraction from text, highlight the potential efficacy of this thesis's LLM approach. Addressing issues like typographical errors, which account for most of the discrepancy in F_1-Scores between PDF and HTML value extraction from text, could therefore significantly enhance performance.

Consequently, while Kamleh's approach currently yields better extraction results, this thesis presents more opportunity for advancement. Several aspects of this new method can be improved, and the ongoing enhancements to the models used by the Assistant API by OpenAI will further increase the capabilities of this thesis's LLM approach in value extraction.

Ultimately, this thesis introduces promising new techniques for data extraction from complex PDF documents, establishing a solid foundation for future research. To enable further development of these methods, I have made the associated code available on GitHub[1] and encourage other researchers to build upon this work, by exploring further applications and refining its methodologies.

[1] https://github.com/lenzen-lysander/bachelor-project

7 FUTURE WORK

With the methods devised in this thesis having been tested and evaluated, this chapter will now talk about the aspects of these methods that still show potential for growth and give input on ways this growth may be achieved.

7.1 EXTRACTION OF TABLES FROM PDF FILES

This thesis identifies the area of tabular data extraction from PDF files as presenting the most opportunities for future enhancements. While the current method successfully extracts tables classified as 'well contained', initial plans also included strategies for integrating the more fragmented 'partially contained' and 'not contained' tables. Incorporating these would likely improve F_1-Scores significantly. However, these plans were not implemented due to the error-prone nature of the method used to determine whether a text section was part of these tables. This method assessed the numeric ratios of sections adjacent to table captions to gauge their likelihood of belonging to a quantitative table. This process was repeated until a section failed the classification, with all preceding sections considered as part of one table. Unfortunately, this thesis discovered that a numeric ratio of 0.15, although effective for extracting 'well contained' tables, was too lenient as a sole criterion for non-tabular data. This issue became particularly evident when attempting to extract 'partially Contained' tables, as the requirement for a minimum number of tokens had to be omitted to correctly identify the smaller disconnected text blocks containing tabular data. Consequently, this made the criteria too easy to satisfy for data that was not genuinely tabular. While intuitively, increasing the minimum numeric ratio might seem like a solution, finding a balance that minimized false positives without excluding too many genuine quantitative tables proved challenging. Although additional extracted text typically does not impair the Assistant API's ability to identify only the intended table, it does detract from text that might be analyzed later. Nevertheless, this thesis posits that an optimal numeric ratio that accurately classifies sections as tabular or textual could potentially be identified.

Alternatively, this thesis proposes that a NLP approach could be developed to classify text blocks based on whether they originate from tabular or textual data. Although this approach was initially explored as it was deemed most suitable for the extraction of 'not contained' tables, by concatenating adjacent text blocks classified as being tabular, it was subsequently abandoned due to the absence of a suitable database for training such a classifier and the excessive time required for its development. Despite these challenges and the lack of assured performance outcomes, it remains another promising direction for future research. It should be mentioned that while both of these methods were devised having in mind one particular table type, both of them could be capable in detecting both 'partially contained' and 'not contained' tables, as well as the 'well contained' ones.

7.2 VALUE EXTRACTION FROM TABLES AND TEXT

The methods developed for value extraction from tables and text also demonstrate significant potential for further refinement. Although their performance already substantially surpasses that of the table extraction, there remain weaknesses that can be addressed. One of the major challenges is the typographical errors introduced during text extraction via OCR. While the Assistant API generally manages to decipher context even with some characters being misinterpreted, the occasional omission of smaller characters, such as commas or asterisks, can lead to the generation of false values and loss of critical information.

Additionally, the false extraction of Greek letters, frequently used to denote mathematical values like effect sizes, presents another hurdle. These characters can typically be extracted using specialized training data provided by Tesseract. However, attempts to implement this training data resulted in a high incidence of false positives and false negatives, leading to its eventual exclusion from this thesis. This thesis believes that these issues could be mitigated by preprocessing the images used as input for pytesseract. Techniques such as gray-scaling, binarization, and others could enhance pytesseract's performance, reducing the number of characters that are incorrectly excluded or misinterpreted.

Another encountered issue involves the incorrect handling of commonly used notations, such as asterisks to indicate significance levels, by the Assistant API. Furthermore, less commonly recognized effect sizes were often not extracted. These problems could be addressed by enriching the prompt for the Assistant API with additional information on common table types and their typical representations of values of interest. Expanding the 'Knowledge' file provided to the Assistant API to include not only types of effect sizes but also their common abbreviations and representations in literature could help the Assistant API more accurately identify a broader range of values and their various notations, thereby enhancing the extraction rate further.
This thesis hopes that the suggested refinements to its methods will inspire future work to further enhance the extraction and analysis of PDF content.

BIBLIOGRAPHY

[1] D. Abrokwa, S. Das, O. Akgul, and M. L. Mazurek. Comparing security and privacy attitudes among {US}. users of different smartphone and {Smart-Speaker} platforms. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*, pages 139–158, 2021.

[2] T. Aitamurto, A. S. Won, S. Sakshuwong, B. Kim, Y. Sadeghi, K. Stein, P. G. Royal, and C. L. Kircos. From fomo to jomo: Examining the fear and joy of missing out and presence in a 360 video viewing experience. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2021.

[3] M. Akter, M. Tabassum, N. S. Miazi, L. Alghamdi, J. Kropczynski, P. J. Wisniewski, and H. Lipford. Evaluating the impact of community oversight for managing mobile privacy and security. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 437–456, 2023.

[4] S. Ali, A. Razi, S. Kim, A. Alsoubai, J. Gracie, M. De Choudhury, P. J. Wisniewski, and G. Stringhini. Understanding the digital lives of youth: Analyzing media shared within safe versus unsafe private conversations on instagram. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2022.

[5] D. G. Balash, D. Kim, D. Shaibekova, R. A. Fainchtein, M. Sherr, and A. J. Aviv. Examining the examiners: Students' privacy and security perceptions of online proctoring services. In *Seventeenth symposium on usable privacy and security (SOUPS 2021)*, pages 633–652, 2021.

[6] D. Bragg, N. Caselli, J. W. Gallagher, M. Goldberg, C. J. Oka, and W. Thies. Asl sea battle: gamifying sign language data collection. In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pages 1–13, 2021.

[7] R. Cheng, S. Dasgupta, and B. M. Hill. How interest-driven content creation shapes opportunities for informal learning in scratch: A case study on novices' use of data structures. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–16, 2022.

[8] A. S. Corrêa and P.-O. Zander. Unleashing tabular content to open data: A survey on pdf table extraction methods and tools. In *Proceedings of the 18th annual international conference on digital government research*, pages 54–63, 2017.

[9] J. Damerow, B. E. Peirson, and M. D. Laubichler. The giles ecosystem–storage, text extraction, and ocr of documents. *Journal of Open Research Software*, 5(1):26–26, 2017.

[10] A. Danilova, A. Naiakshina, J. Deuter, and M. Smith. Replication: On the ecological validity of online security developer studies: Exploring deception in a {Password-Storage} study with freelancers. In *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, pages 165–183, 2020.

[11] P. Emami-Naeini, T. Francisco, T. Kohno, and F. Roesner. Understanding privacy attitudes and concerns towards remote communications during the {COVID-19} pandemic. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*, pages 695–714, 2021.

[12] F. M. Farke, L. Lassak, J. Pinter, and M. Dürmuth. Exploring user authentication with windows hello in a small business environment. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*, pages 523–540, 2022.

[13] E. Gerlitz, M. Häring, and M. Smith. Please do not use!? _ or your license plate number: Analyzing password policies in german companies. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*, pages 17–36, 2021.

[14] X. Han, M. Zhou, M. J. Turner, and T. Yeh. Designing effective interview chatbots: Automatic chatbot profiling and design suggestion generation for chatbot debugging. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–15, 2021.

[15] H. Hazazi and M. Shehab. Exploring the usability, security, and privacy of smart locks from the perspective of the end user. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 559–577, 2023.

[16] F. Herbert, M. Kowalewski, T. Schnitzler, L. Lassak, and M. Dürmuth. {"Fast}, easy,{Convenient."} studying adoption and perception of digital covid certificates. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*, pages 463–482, 2022.

[17] H.-Y. Huang, S. Demetriou, M. Hassan, G. S. Tuncay, C. A. Gunter, and M. Bashir. Evaluating user behavior in smartphone security: a psychometric perspective. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 509–524, 2023.

[18] F. Jahanbakhsh, J. Cranshaw, S. Counts, W. S. Lasecki, and K. Inkpen. An experimental study of bias in platform worker ratings: the role of performance quality and gender. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–13, 2020.

[19] A. Jiju, S. Tuscano, and C. Badgujar. Ocr text extraction. *International Journal of Engineering and Management Research*, 11(2):83–86, 2021.

[20] V. N. S. R. Kamisetty, B. S. Chidvilas, S. Revathy, P. Jeyanthi, V. M. Anu, and L. M. Gladence. Digitization of data from invoice using ocr. In *2022 6th International Conference on Computing Methodologies and Communication (ICCMC)*, pages 1–10. IEEE, 2022.

[21] D. Kamleh. Konvertierung von pdf-dokumenten in maschinenlesbare texte und extraktion von statistischen informationen. 2022.

[22] P. G. Kelley, C. Cornejo, L. Hayes, E. S. Jin, A. Sedley, K. Thomas, Y. Yang, and A. Woodruff. " there will be less privacy, of course": How and why people in 10 countries expect {AI} will affect privacy in the future. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 579–603, 2023.

[23] L. Kersten, T. Mulders, E. Zambon, C. Snijders, and L. Allodi. 'give me structure': Synthesis and evaluation of a (network) threat analysis process supporting tier 1 investigations in a security operation center. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 97–111, 2023.

[24] A. Kitkowska, M. Warner, Y. Shulman, E. Wästlund, and L. A. Martucci. Enhancing privacy through the visual design of privacy notices: Exploring the interplay of curiosity, control and affect. In *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, pages 437–456, 2020.

[25] E. Liu, L. Sun, A. Bellon, G. Ho, G. M. Voelker, S. Savage, and I. N. Munyaka. Understanding the viability of gmail's origin indicator for identifying the sender. In *Nineteenth Symposium on Usable Privacy and Security (SOUPS 2023)*, pages 77–95, 2023.

[26] N. Meuschke, A. Jagdale, T. Spinde, J. Mitrović, and B. Gipp. A benchmark of pdf information extraction tools using a multi-task and multi-domain evaluation framework for academic documents. In *International Conference on Information*, pages 383–405. Springer, 2023.

[27] J. Mink, A. R. Yuile, U. Pal, A. J. Aviv, and A. Bates. Users can deduce sensitive locations protected by privacy zones on fitness tracking apps. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–21, 2022.

[28] R. Mittal and A. Garg. Text extraction using ocr: A systematic review. In *2020 Second International Conference on Inventive Research in Computing Applications (ICIRCA)*, pages 357–362, 2020. doi: 10.1109/ICIRCA48905.2020.9183326.

[29] OpenAI. File search api documentation. . URL https://platform.openai.com/docs/assistants/tools/file-search/quickstart. Last opened on March 3, 2024.

[30] OpenAI. Guide to prompt engineering. . URL https://platform.openai.com/docs/guides/prompt-engineering. Last opened on April 10, 2024.

[31] OpenAI. Create assistant api documentation. . URL https://platform.openai.com/docs/api-reference/assistants/createAssistant. Last opened on March 3, 2024.

[32] K. Owens, O. Anise, A. Krauss, and B. Ur. User perceptions of the usability and security of smartphones as {FIDO2} roaming authenticators. In *Seventeenth Symposium on Usable Privacy and Security (SOUPS 2021)*, pages 57–76, 2021.

[33] V. Paneva, M. Bachynskyi, and J. Müller. Levitation simulator: prototyping ultrasonic levitation interfaces in virtual reality. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–12, 2020.

[34] J. G. Pires. Data science using openai: testing their new capabilities focused on data science. *Qeios*, 2023.

[35] J. Seering, J. Hammer, G. Kaufman, and D. Yang. Proximate social factors in first-time contribution to online communities. In *Proceedings of the 2020 CHI Conference on Human Factors in Computing Systems*, pages 1–14, 2020.

[36] P. Story, D. Smullen, A. Acquisti, L. F. Cranor, N. Sadeh, and F. Schaub. From intent to action: Nudging users towards secure mobile payments. In *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, pages 379–415, 2020.

[37] L. Taber and S. Whittaker. " on finsta, i can say'hail satan'": Being authentic but disagreeable on instagram. In *Proceedings of the 2020 CHI conference on human factors in computing systems*, pages 1–14, 2020.

[38] J. Tang, E. Birrell, and A. Lerner. Replication: How well do my results generalize now? the external validity of online privacy and security surveys. In *Eighteenth symposium on usable privacy and security (SOUPS 2022)*, pages 367–385, 2022.

[39] K. Thomas, P. G. Kelley, S. Consolvo, P. Samermit, and E. Bursztein. "it's common and a part of being a content creator": Understanding how creators experience and cope with hate and harassment online. In *Proceedings of the 2022 CHI conference on human factors in computing systems*, pages 1–15, 2022.

[40] C. Tiefenau, M. Häring, K. Krombholz, and E. Von Zezschwitz. Security, availability, and multiple information sources: Exploring update behavior of system administrators. In *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, pages 239–258, 2020.

[41] D. Wermke, N. Huaman, C. Stransky, N. Busch, Y. Acar, and S. Fahl. Cloudy with a chance of misconceptions: exploring users' perceptions and expectations of security and privacy in cloud office suites. In *Sixteenth Symposium on Usable Privacy and Security (SOUPS 2020)*, pages 359–377, 2020.

[42] T. Whalen, T. Meunier, M. Kodali, A. Davidson, M. Fayed, A. Faz-Hernández, W. Ladd, D. Maram, N. Sullivan, B. C. Wolters, et al. Let the right one in: Attestation as a usable {CAPTCHA} alternative. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*, pages 599–612, 2022.

[43] P. W. Woźniak, J. Karolus, F. Lang, C. Eckerth, J. Schöning, Y. Rogers, and J. Niess. Creepy technology: what is it and how do you measure it? In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pages 1–13, 2021.

[44] W. Yaqub, O. Kakhidze, M. L. Brockman, N. Memon, and S. Patil. Effects of credibility indicators on social media news sharing intent. In *Proceedings of the 2020 chi conference on human factors in computing systems*, pages 1–14, 2020.

[45] B. Yildiz. *Information extraction-utilizing table patterns*. PhD thesis, 2004.

[46] B. Yildiz, K. Kaiser, and S. Miksch. pdf2table: A method to extract table information from pdf files. In *IICAI*, volume 2005, pages 1773–1785. Citeseer, 2005.

[47] Y. Zhang, N. Suhaimi, N. Yongsatianchot, J. D. Gaggiano, M. Kim, S. A. Patel, Y. Sun, S. Marsella, J. Griffin, and A. G. Parker. Shifting trust: Examining how trust and distrust emerge, transform, and collapse in covid-19 information seeking. In *Proceedings of the 2022 CHI Conference on Human Factors in Computing Systems*, pages 1–21, 2022.

[48] V. Zhao, L. Zhang, B. Wang, M. L. Littman, S. Lu, and B. Ur. Understanding trigger-action programs through novel visualizations of program differences. In *Proceedings of the 2021 CHI Conference on Human Factors in Computing Systems*, pages 1–17, 2021.

[49] S. Zheng and I. Becker. Presenting suspicious details in {User-Facing} e-mail headers does not improve phishing detection. In *Eighteenth Symposium on Usable Privacy and Security (SOUPS 2022)*, pages 253–271, 2022.

APPENDIX

Listing A.1: Prompt for extracting statistical data from text.

```
'''You will receive a message with an attached text file containing an excerpt from a scientific paper's results section.
Please follow these instructions:

1. Thoroughly review the entire excerpt.
2. Identify and extract all p-values, confidence intervals (CIs), and effect sizes mentioned in the text. These often appear in pairs.
3. Note that:
   - p-values usually represent significance levels at or below 0.05.
   - CIs are usually enclosed in square brackets and preceded by 'CI'.
   - For effect sizes consider those mentioned in your 'knowledge' file,
     they may look unusual due to the incorrect representation of Greek letters in the text.
     Be mindful of the context to ensure no values are overlooked.
4. Exclude any values that appear to be part of tables or other structured elements rather than in the narrative text.
5. For each statistic, check if it correlates with another (e.g., a p-value with a corresponding effect size or CI).
6. If a statistic appears more than once, record each occurrence.
7. Output your findings using the following template:

Output Template:
"p-values:
p_value1
p_value2
...

effect sizes:
es_1
es_2
...

confidence intervals:
ci_1
ci_2
...

Example output:
"p-values:
p = 0.01
p < 0.001
p < 0.05

effect sizes:
OR = 6.51
W = 32.5
r = 0.08

confidence intervals:
[1.51, 33.18]
[0.04, 2.69]
[0.03, 3.87]"

If no values are found, use the following format:
"p-values:
No p-values found.

effect sizes:
No effect sizes found.

confidence intervals:
No intervals found."

Ensure that your response strictly adheres to the output template without additional commentary.'''
```

Listing A.2: Prompt for extracting statistical data from tables.

```
'''You will receive a message containing one or more tables, each separated by 40 hyphens.
Please follow these instructions:

1. Review each table in its entirety.
2. Identify and extract all p-values, confidence intervals (CIs), and effect sizes mentioned in each table. These often appear in pairs.
3. Note that:
   - p-values are typically labeled as 'p-value' or 'p' and represent significance levels at or below 0.05.
   - CIs are usually enclosed in square brackets and labeled 'CI'.
   - For effect sizes, consider those mentioned in your 'knowledge' file,
     they might be labeled with terms like 'coefficient', 'odds ratio', among others,
     and may appear unusual due to the incorrect representation of Greek letters. Pay attention to the context to avoid missing any values.
4. For each statistic, check if it correlates with another (e.g., a p-value with a corresponding effect size or CI).
5. Record each statistic occurrence, even if it appears multiple times.
6. Output your findings using the following template:

Output Template:
"Table 1:
p-values:
p_value1
p_value2
...

effect sizes:
es_1
es_2
...

confidence intervals:
ci_1
ci_2
...

Table 2:
..."

Example Output:
"Table 1:
p-values:
p = 0.01
p < 0.001
p < 0.05

effect sizes:
OR = 6.51
W = 32.5
r = 0.08

confidence intervals:
[1.51, 33.18]
[0.04, 2.69]
[0.03, 3.87]

Table 2:
..."

If no values are found in a table, use the following format:
"Table X:
p-values:
No p-values found.

effect sizes:
No effect sizes found.

confidence intervals:
No intervals found."

Ensure that your response strictly adheres to the output template without additional commentary.'''
```

Table A.1: Detailed extraction results from the test on SOUPS papers in PDF format

Examined Paper	Numeric Tables			Table Values*			Text Values*		
	Present	TP	FP	Present	TP	FP	Present	TP	FP
Danilova et al. [10]	6	5	0	21,9,7	16,7,5	0,0,0	13,9,2	9,5,2	0,0,0
Kitkowska et al. [24]	6	4	0	38,53,0	20,26,0	0,0,0	25,22,3	19,14,3	0,1,0
Story et al. [36]	6	6	1	55,43,0	51,41,0	0,0,0	9,17,0	8,16,0	3,0,0
Tiefenau et al. [40]	5	5	0	0,0,0	0,0,0	0,0,0	13,2,0	13,2,0	0,0,0
Wermke et al. [41]	2	2	0	4,0,4	4,0,4	0,0,0	9,0,0	9,0,0	2,0,0
Abrokwa et al. [1]	5	4	0	28,28,28	10,10,10	0,0,0	10,0,0	8,0,0	9,0,14
Balash et al. [5]	6	4	1	96,25,0	47,21,0	0,0,0	8,8,0	8,8,0	0,0,0
Emami-Naeini et al. [11]	2	0	0	23,23,0	9,7,0	0,0,0	8,7,0	8,7,0	0,0,0
Gerlitz et al. [13]	3	3	1	0,0,0	0,0,0	0,0,0	4,3,0	4,3,0	0,0,0
Owens et al. [32]	5	5	0	9,9,9	9,6,9	0,0,0	14,13,0	12,10,0	1,0,0
Farke et al. [12]	1	0	0	6,6,0	0,0,0	0,0,0	2,0,0	2,0,0	5,6,0
Herbert et al. [16]	4	2	0	4,13,0	0,0,0	0,0,0	2,0,0	2,0,0	0,0,0
Tang et al. [38]	5	5	0	36,36,0	14,34,0	0,0,0	2,19,0	2,16,0	0,3,0
Whalen et al. [42]	0	0	0	0,0,0	0,0,0	0,0,0	1,1,0	1,1,0	0,0,0
Zheng et al. [49]	4	4	0	26,26,0	25,25,0	0,0,0	32,41,0	23,29,0	0,10,0

* P-Values, effect sizes, confidence intervals.

Table A.2: Detailed extraction results from the test on CHI papers in PDF format

Examined Paper	Numeric Tables			Table Values*			Text Values*		
	Present	TP	FP	Present	TP	FP	Present	TP	FP
Jahanbakhsh et al. [18]	1	0	0	8,8,8	0,0,0	0,0,0	28,17,17	28,17,17	8,8,8
Paneva et al. [33]	1	0	0	0,0,0	0,0,0	0,0,0	7,1,0	6,0,0	0,0,0
Seering et al. [35]	16	13	0	111,111,0	94,0,0	0,0,0	0,0,0	0,0,0	0,0,0
Taber et al. [37]	2	2	0	14,17,12	14,10,12	1,0,0	1,1,0	1,0,0	0,0,0
Yaqub et al. [44]	3	2	0	20,20,20	20,20,20	0,0,0	15,15,4	15,14,2	0,0,0
Aitamurto et al. [2]	5	2	0	10,0,0	0,0,0	0,0,0	16,14,0	10,0,0	0,0,0
Bragg et al. [6]	7	4	0	6,0,0	6,0,0	0,0,0	3,2,0	3,0,0	0,0,0
Han et al. [14]	3	3	0	24,24,0	22,0,0	0,0,0	1,2,0	1,2,0	0,0,0
Woźniak et al. [43]	4	2	0	4,0,0	0,0,0	0,0,0	1,1,1	1,0,1	4,0,0
Zhao et al. [48]	1	1	0	19,0,0	17,0,0	0,0,0	24,6,0	23,0,0	0,0,0
Ali et al. [4]	4	3	0	0,0,0	0,0,0	0,0,0	10,4,0	10,4,0	0,0,0
Cheng et al. [7]	3	0	0	12,12,0	0,0,0	0,0,0	6,6,0	6,6,0	2,2,0
Mink et al. [27]	3	2	1	4,6,0	3,6,0	0,0,0	9,10,0	9,3,0	0,0,0
Thomas et al. [39]	5	4	0	27,27,0	27,27,0	0,0,0	8,3,0	7,3,0	0,0,0
Zhang et al. [47]	2	2	1	18,34,0	3,16,0	0,0,0	17,17,0	16,15,0	0,0,0

* P-Values, effect sizes, confidence intervals.

Table A.3: Detailed extraction results from the test on CHI papers in HTML format

Examined Paper	Numeric Tables			Table Values*			Text Values*		
	Present	TP	FP	Present	TP	FP	Present	TP	FP
Jahanbakhsh et al. [18]	1	1	0	8,8,8	8,8,8	0,0,0	28,17,17	24,16,16	0,0,0
Paneva et al. [33]	1	1	0	0,0,0	0,0,0	0,0,0	7,1,0	7,0,0	0,0,0
Seering et al. [35]	16	16	0	111,89,0	89,89,0	0,15,0	0,0,0	0,0,0	0,0,0
Taber et al. [37]	2	2	0	14,17,12	12,17,12	1,0,0	1,1,0	1,1,0	0,0,0
Yaqub et al. [44]	3	3	0	20,20,20	20,20,20	0,0,0	15,15,4	14,15,4	0,0,0
Aitamurto et al. [2]	5	5	0	10,0,0	10,0,0	0,0,0	16,14,0	13,14,0	0,0,0
Bragg et al. [6]	7	7	0	6,0,0	6,0,0	0,0,0	3,2,0	3,0,0	0,0,0
Han et al. [14]	3	3	2	24,24,0	13,22,0	0,0,0	1,2,0	1,2,0	0,0,0
Woźniak et al. [43]	4	4	0	4,0,0	4,0,0	0,0,0	1,1,1	1,1,1	0,0,0
Zhao et al. [48]	1	1	0	19,0,0	19,0,0	0,0,0	24,6,0	24,4,0	2,0,0
Ali et al. [4]	4	4	0	0,0,0	0,0,0	0,0,0	10,4,0	9,4,0	2,2,0
Cheng et al. [7]	3	3	0	12,12,0	10,0,0	0,0,12	6,6,0	6,6,0	0,0,0
Mink et al. [27]	3	3	0	4,6,0	4,4,0	0,0,0	9,10,0	9,8,0	0,0,0
Thomas et al. [39]	5	5	0	27,27,0	27,27,0	0,0,0	8,3,0	6,3,0	0,0,0
Zhang et al. [47]	2	2	0	18,34,0	16,34,0	0,0,0	17,17,0	14,17,0	0,0,0

* P-Values, effect sizes, confidence intervals.