CACHE-ED: Redefining Document Entity Extraction with Graph-Based Templates, Actor-Critic Agents & HIL

Sudhanshu Bhoi*
Amazon
sudhbee@amazon.com

Harish Y V S*
Amazon
yvsharis@amazon.com

Abstract

In this paper, we present CACHE-ED, a novel framework for document entity extraction that combines the power of large language models (LLMs) with graph-based document representations, caching mechanisms, and an actor-critic multi-agent architecture. Our approach addresses the inefficiencies and inaccuracies that are common in extracting structured information from documents, particularly in templated formats like invoices. CACHE-ED implements a human-in-the-loop paradigm in which human reviewers validate and update the system's outputs, establishing a feedback loop that progressively enhances the accuracy of future extractions. This process ultimately eliminates the need for human intervention over time and optimizes tail-end accuracies, refining the system from 90% to near-perfect precision. Our experiments demonstrate that this approach outperforms industry wide used extraction mechanisms by 8% and improves the speed & reduces cost by over 50% each, making it a scalable solution for real-world applications.

Keywords

Document Entity Extraction, Graph-based Templates, Actor-Critic Systems, Large Language Models, Caching Mechanisms, Human-in-the-Loop Learning, Document Processing, Information Extraction, Template Matching, Data Validation

ACM Reference Format:
Sudhanshu Bhoi and Harish Y V S. 2025. CACHE-ED: Redefining Document Entity Extraction with Graph-Based Templates, Actor-Critic Agents & HIL. In *Proceedings of KDD 2025 workshop on AI Agent for Information Retrieval (Agent4IR @ KDD 2025).* ACM, New York, NY, USA, 7 pages. https://doi.org/XXXXXXX.XXXXXXX

1 Introduction

Document entity extraction has become a crucial task across industries, from invoice processing in finance to legal document analysis, where the accurate identification of key information is essential. Despite significant advances in natural language processing (NLP) and large language models (LLMs), current state-of-the-art systems still struggle to deliver high accuracy, particularly achieving 100% accuracy on challenging documents from **Day 1** remains unattainable, even with the significant advancements in current technologies, making it difficult to optimize "tail-end" accuracy. Tools like AWS Textract, Google Cloud Document AI, and LLMs such as GPT-based models or Anthropic's Claude demonstrate strong baseline performance, but they fall short in complex, templated document formats or edge cases, where high precision is needed.

A major challenge with these systems is their inconsistency in handling structured documents that have recurring but slightly varied templates, often leading to the incorrect extraction of one or two fields. For instance, invoices, contracts, or forms that follow similar patterns but differ in layout can confuse even the most advanced models, resulting in errors or incomplete extractions. Moreover, while LLMs like Claude and GPT-4 are excellent at understanding and generating language, they are computationally expensive and often require repeated calls to process documents that share similar structures & yet do not guarantee results that are replicable. This not only increases costs but also leads to inefficiencies in workflows, particularly when processing large volumes of documents in real-time environments.

The stakes of such minor misses/inaccurate extraction are higher in financial documents, an error in extracting or validating tax amounts, net totals, or country-specific VAT rates can cause significant downstream issues. Current systems often lack robust mechanisms for validating extracted information against external compliance standards or cross-referencing with domain-specific data, leaving room for errors in critical areas.

Moreover, existing LLM-based extraction tools do not inherently learn from their mistakes. Although human reviewers are often needed to correct errors, this feedback is rarely integrated into the system in a meaningful way, leading to repeated mistakes and inefficiencies over time. As a result, organizations often face the dual challenge of handling high costs and inconsistent accuracy, especially for recurring document formats.

In response to these challenges, in this work, we propose **CACHE-ED** (**C**ached & **A**daptive decisions that are validated by **C**ritic & learned from **H**uman-in-the-loop to perform **E**ntity **E**xtraction in **D**ocument), a novel framework designed to overcome these limitations by combining graph-based document representation, caching for recurring templates, and an actor-critic LLM model with external validation tools. Our approach not only improves the efficiency of document entity extraction but also introduces a self-learning mechanism that continuously improves tail-end accuracy, ensuring that errors are corrected and not repeated.

Our contributions are summarised as follows:

- We introduce a novel document graph representation that leverages existing OCR technologies to capture the semantic

*Equal contribution.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. Copyrights for components of this work owned by others than the author(s) must be honored. Abstracting with credit is permitted. To copy otherwise, or republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee. Request permissions from permissions@acm.org.
Agent4IR @ KDD 2025, Toronto, CA
© 2025 Copyright held by the owner/author(s). Publication rights licensed to ACM.
ACM ISBN 978-1-4503-XXXX-X/2025/08
https://doi.org/XXXXXXX.XXXXXXX

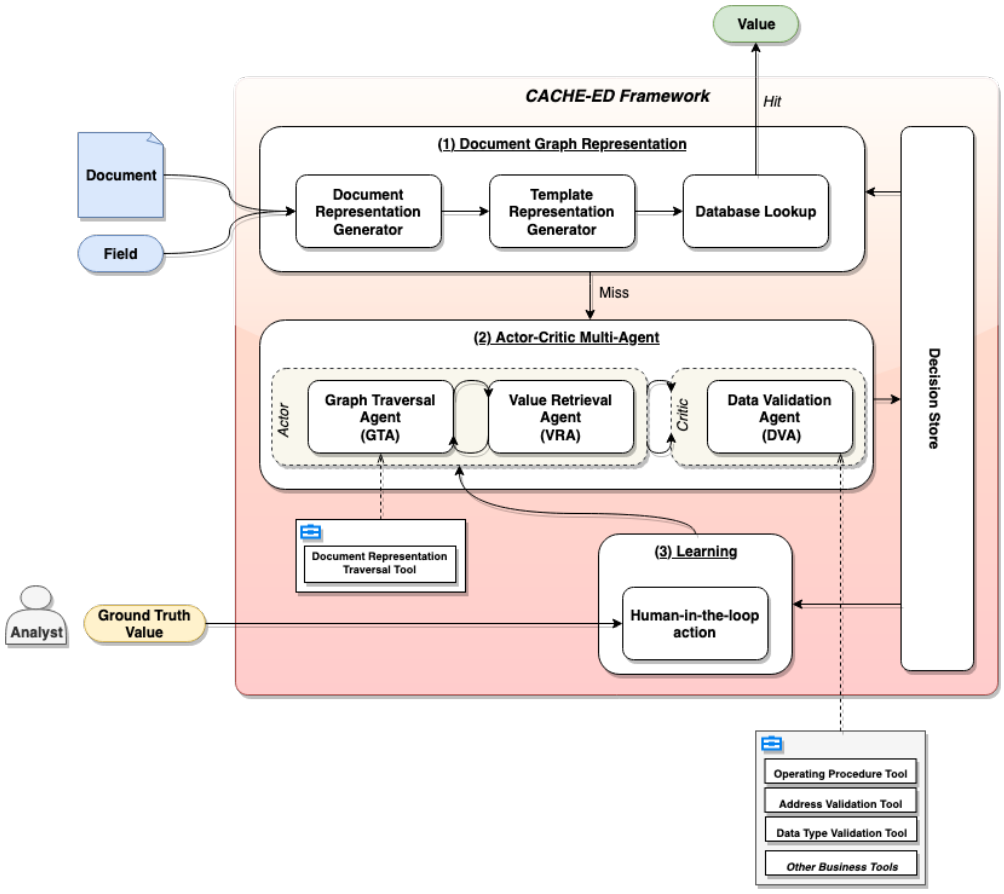

Figure 1: Overview of the CACHE-ED Framework. The process begins with Graph Representation. This is followed by the Actor-Critic Multi-Agent Architecture, where the Actor extracts data through graph traversal and the Critic validates the output using external tools. The framework incorporates Learning via Human-in-the-Loop, where human feedback refines the extraction (if required), and results are stored in a Decision Store.

relationships between various entity relations within the document.

- We introduce the concept of caching graph template representations for incoming documents to lower data extraction costs and improve processing speed.
- We present a comprehensive benchmark for document data extraction, comparing various existing approaches and demonstrating the effectiveness of our caching strategy. Our results show state-of-the-art performance in this extensive evaluation.

2 Related Work

Document information extraction techniques using Large Language Models (LLMs) can be categorized into white-box and black-box approaches. White-box techniques such as UIPath [4], TextMonkey [8], or DocLLM [13], require fine-tuning of LLMs and access to internal document representations, which can be expensive and infeasible when using proprietary LLM APIs. Conversely, black-box techniques treat LLMs as closed systems and have been explored for extracting information from legal contracts [15] and research papers [10] using semantic parsing and prompt engineering. However, these methods are optimized for retrieving specific sets of fields [11] and, as observed in our experiments, fall short due to a lack of entity-relation understanding.

First introduced in [9], extraction of information from documents as entities and their relationships has been possible with use of visual and textual encoder and decoder networks [7] [12]. However, these representations contain redundant links, lack context links, partially identify keys and fail to tag values in a line. Our approach overcomes these problems by building document representation on top of AWS Textract's [2] output and use our CACHE-ED framework to map information to any business-defined fields.

LLM-based solutions raise concerns regarding reliability, accuracy, performance, and cost-efficiency. Several solutions have been explored to address these problems, including external tool validation [5], learning from human-in-the-loop interactions [14], and

caching of prompts and answers [6] [3]. Our CACHE-ED framework incorporates these techniques to ensure reliable, accurate, and cost-effective document information extraction.

3 Approach

In this work, we introduce a novel framework for document entity extraction that leverages large language models (LLMs) in conjunction with graph-based document representations, caching mechanisms, and actor-critic multi-agent architecture. Our approach is designed to improve efficiency, accuracy, and self-learning capability in structured document extraction tasks, particularly for documents that follow repetitive or templated formats such as invoices. The methodology centers on four key innovations: converting documents into a graph structure, caching extraction processes for reuse, using an actor-critic LLM multi-agent framework, and introducing human in the loop actions for continuously improving the system over time. The complete end to end flow is show in Figure 1 & refer the algorithm 1 for the control/data flow of the framework.

3.1 Graph-Based Representation

We leverage Block objects and relationships provided by AWS Textract [2] (refer Figure 2 graph (A)) to analyze the document structure and content. We extract information that was missed by the KEY_VALUE_SET Block from each of the LINE Block using Named Entity Recognition (NER) to identify and mask Key and pair it with associated Value (3). We then distill the information from nested structures into logical parent-child Key-to-Key relations. For example, the TABLE Block is represented as parent Key with table information, which is connected to multiple child Keys representing column headers.

To ensure that our representation is robust to document orientations, variations in the page dimensions, and length of intermediary structures, we aggregate all of the PAGE blocks under a single Document node. Furthermore, we incorporate spatial and contextual awareness between the child Keys of each node by introducing adjacent Key-to-Key relations using a proximity-based equation 3. Finally, we anonymize the node identifiers so that paths to Values can be used across different document instances of the same template. The resulting graph is referred to as the document representation graph (refer Figure 2 graph (B)). By removing the Value nodes, which are the only variable nodes across different document instances of a template, we obtain the template representation graph (refer Figure 2 graph (C)).

With these graph representations, the system can effectively model the relationships between different entities and accommodate even complex document layouts. When an input query with field is provided, the actor (section 3.3) traverses the document representation graph to identify relevant nodes for data extraction.

$$DR_{page}(x, y, p) = (x, p + y - 1) \tag{1}$$

$$P(\alpha, \beta) = D_{euclid}(DR_{page}(x^{\alpha}, y^{\alpha}, p^{\alpha}), DR_{page}(x^{\beta}, y^{\beta}, p^{\beta})) \tag{2}$$

$$L(\alpha, \beta) = signum(D_{thresh} - P(\alpha, \beta)) \tag{3}$$

Where

- x, y represent coordinates and p represents page of a bounding box
- α, β represent two candidate entity bounding boxes
- DR_{page} is a page dimension reduction function that maps 3D to 2D space
- P is a proximity function that calculates euclidean distance in the mapped coordinate
- L is a linking function that determines whether link between the two entity is to be established based on threshold distance D_{thresh}

3.2 Caching mechanism for Repeated Templates

To reduce redundant LLM computations for recurring documents of the same template, we introduce a caching mechanism that associates the template graph representation of previously processed documents T_r and field F with chosen path to corresponding value P_i. When a new document arrives, its template representation T_r and query field F are compared with those of previously processed documents. If a match is found, the system bypasses the need for another LLM call and directly retrieves the stored information. The retrieved path to value P_i is then used to traverse the document representation D_r and find required value V.

This cache-based approach significantly speeds up the extraction process for recurring templates, making it particularly cost & speed effective in use cases like invoice processing, where formats are often consistent across multiple documents from the same source. Moreover, it enables the system to learn information for one document and apply it to all documents with the same template (section 3.4).

3.3 Actor-Critic Multi-Agent Module

This module serves as the core of the CACHE-ED framework, composed of two primary components: (i) the Actor, which includes the Graph Traversal Agent (GTA) and the Value Retrieval Agent (VRA), and (ii) the Critic, which houses the Data Validation Agent (DVA). The Actor and Critic modules are designed to be extensible, allowing for the inclusion of additional agents to enhance extraction capabilities based on document complexity.

The Graph Traversal Agent (GTA) is responsible for identifying multiple candidate paths P that are relevant to the field F being extracted from the document representation D_r. GTA utilizes the graph traversal tool to explore the neighboring nodes for more context or to find the initial set of Key nodes based on semantics of the field F. The Value Retrieval Agent (VRA) then selects the correct path P_j, where the terminal node contains the value V for the field F.

Once the Actor has performed the extraction, the Critic, consisting of Data Validation Agent (DVA), steps in to validate the extracted data using external tools. For example, the Critic can call APIs such as the Amazon Location API to verify address fields like "Ship from", "Ship to" or "Bill to" & verify the acceptable VAT rates by identifying the country of origin, or it can perform arithmetic checks to ensure that the net amount plus tax equals the gross amount, or conduct data type, regex & other rule-based validations. This combination of extraction and validation ensures higher accuracy and reduces errors in the data retrieval process.

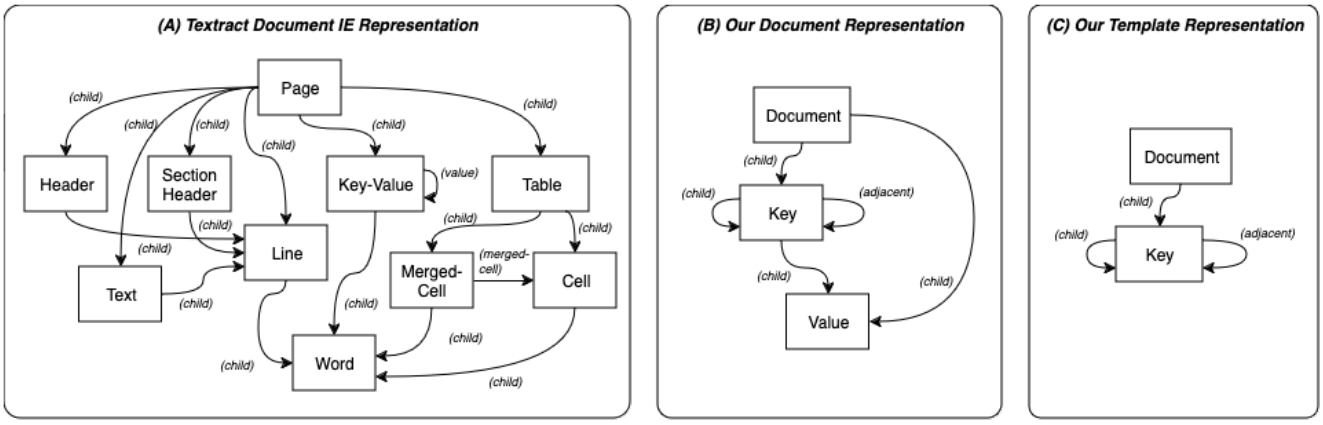

Figure 2: Three states of document representation. Part (A) shows AWS Textract Blocks with relations. Part (B) shows document representation D_r retaining required information into smaller structure with context aware links. Part (C) shows template representation T_r without document-instance specific value nodes.

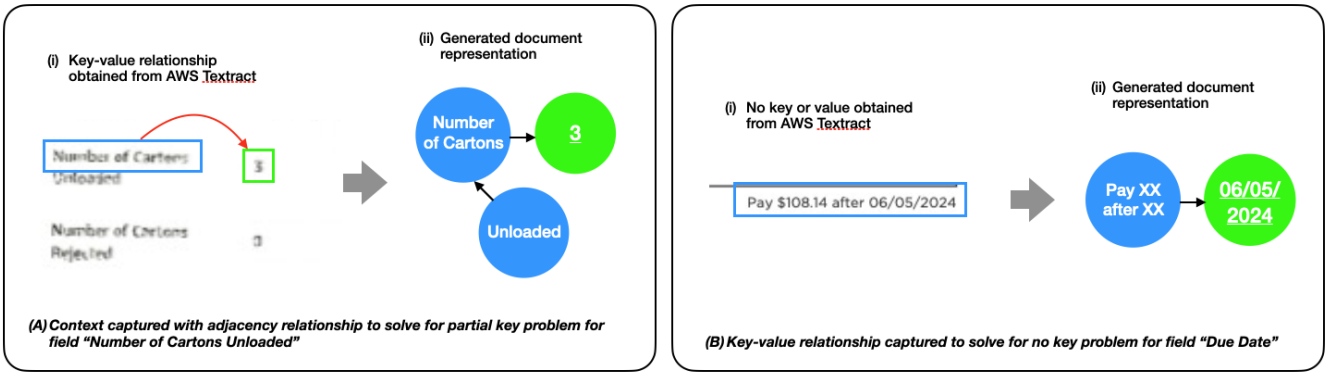

Figure 3: Two cases of key-value relation capture miss by AWS Textract. Part (A) shows the case of partial key capture and how the document representation overcomes it with context-aware links. Part (B) shows the case of no key capture and how developing new links between masked key and value overcomes this.

3.4 A Self-Learning System with Human-in-the-Loop

Human reviewers play an integral role in refining the system's accuracy over time. After the extraction and validation processes, human reviewers are tasked with reviewing the results and making corrections if necessary. The corrected value V_{HIL} from the reviewer are passed as feedback to the Actor (section 3.3), which determines the new path to value P_k. Subsequently, the path in the cache is updated from P_j to P_k, ensuring that new documents of same template retrieve accurate results in future without requiring further human intervention or LLM calls.

This human-in-the-loop feedback mechanism creates a continuously improving system, preventing the repetition of errors by the LLM and ensuring that the system becomes more robust and accurate over time. It further enables the framework to adapt to complex document structures and edge cases, while gradually reducing the need for human intervention as the system learns and refines its extraction capabilities.

4 Experiments

To evaluate the performance of our proposed CACHE-ED framework, we conducted a comprehensive comparison with state-of-the-art document extraction techniques. These techniques included the current solution of AWS Textract [2] with key to field exact-match (Textract + Key EM) mappings, a Multimodal LLM [1], and a hybrid approach that passes AWS Textract ouptut to LLM without any intermediate representations (Textract + LLM). The goal of the experiment was to measure the effectiveness of CACHE-ED in terms of accuracy, efficiency, and repeatability when extracting structured information from invoices of varying complexity.

Comparison across approaches on various invoice types in percentage (%)				
Approaches (→)	Textract + Key EM	Multimodal LLM	Textract + LLM	**CACHE-ED**
Vendor Name	57.9	73.69	77.78	**84.21**
Customer Number	38.24	50	55	**60**
Invoice Number	66.05	75	70	**77.78**
Invoice Date	86	**94.73**	83.33	94.45
Invoice Amount	83.23	**89**	**89**	**89**

Table 1: We present quantitative results across different approaches on medium and hard invoice samples on some of the keys as shown above. For our Actor-Critic agent architecture, we utilized the Claude3-Sonnet model. We see that CACHE-ED performs at par with other approaches.

Performance analysis across various Claude family models in percentage (%)			
Approaches (→)	ClaudeV3-Sonnet	ClaudeV3-Haiku	ClaudeV3-Opus
Vendor Name	85	85	85
Customer Number	60	50	60

Table 2: Table illustrating how CACHE-ED remains agnostic to the Claude 3 family of models. Indicating the effectiveness of our document representation and actor agent.

Algorithm 1 CACHE-ED - Document Field Retrieval Algorithm

Require: Document D, Field F
$D_r \leftarrow$ GenerateDocumentRepresentation(D) ▷ Document Graph Representation
$T_r \leftarrow$ GenerateTemplateRepresentation(D_r)
$P_i \leftarrow$ DBLookup(T_r, F)
if P_i is present **then**
  $V \leftarrow$ GetValueForDocument(D_r, P_i)
  return Value V
else ▷ Actor-Critic Multi-Agent
  do
    Invoke Graph Traversal Agent (GTA): $P \leftarrow$ GTA(D_r, F)
    Invoke Value Retrieval Agent (VRA): $P_j, V \leftarrow$ VRA(P, F)
    Invoke Data Validation Agent (DVA): Valid $\leftarrow$ DVA(V, F)
  while Valid is False & V is present
  if V is present **then**
    DBCreate(T_r, F, P_j)
    return Value V
  end if
end if
$V_{HIL} \leftarrow$ Value from Human-in-the-Loop (HIL) ▷ Learning
if V_{HIL} is not equal to V **then**
  do
    Invoke Graph Traversal Agent (GTA): $P \leftarrow$ GTA(D_r, F, V_{HIL})
    Invoke Value Retrieval Agent (VRA): $P_k, V \leftarrow$ VRA(P, F, V_{HIL})
  while V is present & V is not equal to V_{HIL}
  if V is present **then**
    DBUpdate(T_r, F, P_k)
  end if
end if

We evaluated our approach using a subset of real-world invoices from the RVL-CDIP dataset, which we categorized into three distinct levels of extraction difficulty.:

- **Easy**: The easiest level comprised invoices with a simple, well-aligned structure, minimal variation in formatting, & limited textual content..
- **Medium**: Scanned Invoices with more complex layouts, including tables and varying formats for key-value pairs.
- **Hard**: Scanned Invoices with highly unstructured, unclear & cluttered formats, hand written/scanned, varied layouts, and irregular/random placement of key-value information.

The performance of each approach was evaluated based on accuracy, defined as percentage of correctly extracted fields, across the three complexity levels mentioned above. As all four approaches performed equally well on easy category invoices, we report extraction accuracy metrics only on the medium and hard categories for the quantitative comparison in Table 2.

We further analysed the performance of the CACHE-ED framework across three LLMs of the Claude 3 family [1] with varying order of capability. This analysis helped us measure the efficacy of the proposed document representation D_r and the actor agent (section 3.3). As demonstrated in the Table 3, the CACHE-ED framework remained agnostic to the capabilities of the LLM and maintained consistent performance.

We also present a quantitative analysis of cache hit percentages for a group of vendors in Table 3. By utilizing the proposed template representation T_r and caching mechanism, we achieve at least a 50% reduction in the costs associated with LLM. Further, this approach significantly improves latency and ensures repeatability, which becomes a substantial advantage as the system scales. It is important to note that the dataset is restricted by a small timeframe, and the numbers are expected to improve over time.

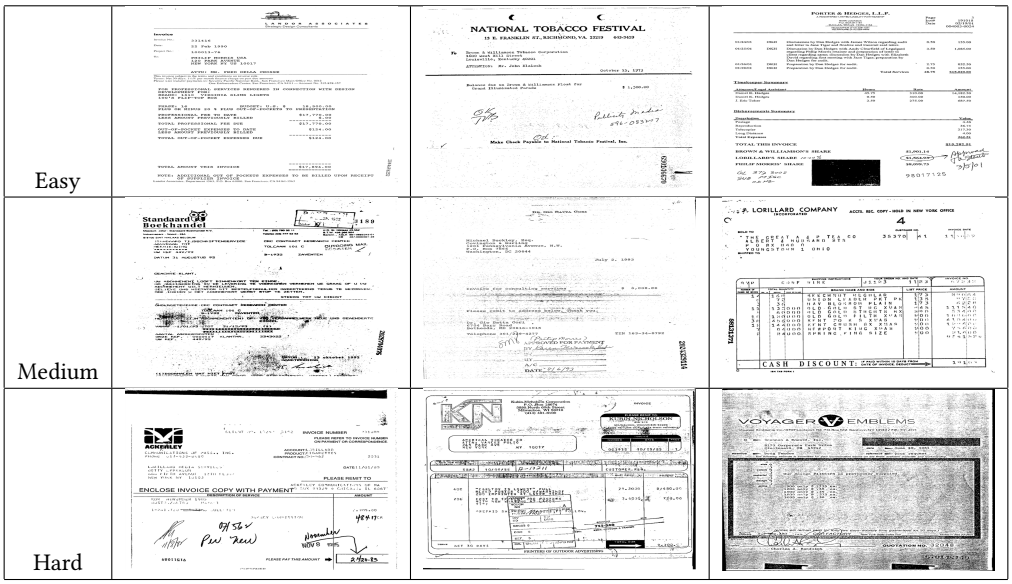

Figure 4: Sample real-world invoices from the RVL-CDIP dataset

Vendors	Cache Hit %
A	58.33
B	70.58
C	54.54
D	56.25
E	60
F	88.89

Table 3: Table demonstrating the effectiveness of our caching strategy: Cache hit percentages across various vendor templates.

Additionally, the DVA (data validation agent) in the actor-critic multi-agent (section 3.3) allowed CACHE-ED to incorporate external validation tools, such as compliance checks for VAT rates and gross amount calculations, improving extraction accuracy for complex invoices. In contrast, other methods mentioned above lead to inaccuracies in critical fields like tax amounts and total sums, particularly in hard invoice templates.

Overall, CACHE-ED demonstrated superior performance across all categories, with a notable advantage in medium and hard invoices where traditional systems faced the greatest challenges. The novel graph-based document representations, caching mechanism, and incorporation of external validation tools contribute to the improved accuracy, repeatability, and efficiency of the CACHE-ED framework in document entity extraction tasks.

5 Conclusion

In this paper, we introduced CACHE-ED, a novel document entity extraction framework that combines graph-based representations, an actor-critic architecture, and a caching mechanism to optimize accuracy, reduce costs, and enhance efficiency. By caching document templates and incorporating human-in-the-loop feedback, CACHE-ED continuously improves extraction performance, particularly for recurring templates. For invoice document extraction, our approach outperformed existing methods, especially on complex templates, while reducing extraction errors and improving validation. Experiments demonstrated the effectiveness of our caching strategy in lowering latency and cost, showcasing the framework's adaptability and scalability across various document complexities.

References

[1] Anthropic. 2024. Introducing the next generation of Claude. https://www.anthropic.com/news/claude-3-family
[2] AWS. 2022. What is Amazon Textract? https://aws.amazon.com/textract/
[3] Fu Bang. 2023. GPTCache: An Open-Source Semantic Cache for LLM Applications Enabling Faster Answers and Cost Savings. 212–218. doi:10.18653/v1/2023.nlposs-1.24
[4] Aaron Crossey Blog. [n. d.]. DocPath: A fine-tuned large language model for information extraction from documents. https://www.uipath.com/blog/ai/docpath-information-extraction-large-language-model
[5] Zhibin Gou, Zhihong Shao, Yeyun Gong, Yelong Shen, Yujiu Yang, Nan Duan, and Weizhu Chen. 2024. CRITIC: Large Language Models Can Self-Correct with Tool-Interactive Critiquing. arXiv:2305.11738 [cs.CL] https://arxiv.org/abs/2305.11738
[6] Jiaxing Li, Chi Xu, Feng Wang, Isaac M von Riedemann, Cong Zhang, and Jiangchuan Liu. 2024. SCALM: Towards Semantic Caching for Automated Chat Services with Large Language Models. arXiv:2406.00025 [cs.CL] https://arxiv.org/abs/2406.00025
[7] Haofu Liao, Aruni RoyChowdhury, Weijian Li, Ankan Bansal, Yuting Zhang, Zhuowen Tu, Ravi Kumar Satzoda, R. Manmatha, and Vijay Mahadevan. 2023. DocTr: Document transformer for structured information extraction in documents. In *ICCV 2023*. https://www.amazon.science/publications/doctr-document-transformer-for-structured-information-extraction-in-documents
[8] Yuliang Liu, Biao Yang, Qiang Liu, Zhang Li, Zhiyin Ma, Shuo Zhang, and Xiang Bai. 2024. TextMonkey: An OCR-Free Large Multimodal Model for Understanding Document. arXiv:2403.04473 [cs.CV] https://arxiv.org/abs/2403.04473
[9] Berke Oral, Erdem Emekligil, Seçil Arslan, and Gülşen Eryiğit. 2020. Information Extraction from Text Intensive and Visually Rich Banking Documents. *Information Processing Management* 57, 6 (2020), 102361. doi:10.1016/j.ipm.2020.102361
[10] Maciej P. Polak and Dane Morgan. 2024. Extracting accurate materials data from research papers with conversational language models and prompt engineering. *Nature Communications* 15, 1 (2 2024). doi:10.1038/s41467-024-45914-8

[11] Filip Seitl, Tomáš Kovářík, Soheyla Mirshahi, Jan Kryštůfek, Rastislav Dujava, Matúš Ondreička, Herbert Ullrich, and Petr Gronat. 2024. Assessing the quality of information extraction. arXiv:2404.04068 [cs.CL] https://arxiv.org/abs/2404.04068

[12] Jianqiang Wan, Sibo Song, Wenwen Yu, Yuliang Liu, Wenqing Cheng, Fei Huang, Xiang Bai, Cong Yao, and Zhibo Yang. 2024. OmniParser: A Unified Framework for Text Spotting, Key Information Extraction and Table Recognition. arXiv:2403.19128 [cs.CV] https://arxiv.org/abs/2403.19128

[13] Dongsheng Wang, Natraj Raman, Mathieu Sibue, Zhiqiang Ma, Petr Babkin, Simerjot Kaur, Yulong Pei, Armineh Nourbakhsh, and Xiaomo Liu. 2023. DocLLM: A layout-aware generative language model for multimodal document understanding. arXiv:2401.00908 [cs.CL] https://arxiv.org/abs/2401.00908

[14] Hengjia Xiao and Peng Wang. 2024. LLM A*: Human in the Loop Large Language Models Enabled A* Search for Robotics. arXiv:2312.01797 [cs.RO] https://arxiv.org/abs/2312.01797

[15] Yu Zhao and Haoxiang Gao. 2024. Utilizing Large Language Models for Information Extraction from Real Estate Transactions. arXiv:2404.18043 [cs.CL] https://arxiv.org/abs/2404.18043