Privacy-Preserving Information Extraction with Local LLMs: A Comparative Study on Dutch Debt Collection Letters

Beyza Celep, Natalia Amat-Lefort, Joost Visser
Leiden Institute of Advanced Computer Science
Leiden University, The Netherlands
b.celep@umail.leidenuniv.nl, {n.amat.lefort, j.m.w.visser}@liacs.leidenuniv.nl

Abstract

For individuals in financial distress, understanding debt collection letters is critical. These documents are often unstructured, use complex legal language, and contain highly sensitive personal data. Automating information extraction is essential for assisting caseworkers, who currently perform this task manually; a slow and error-prone process. The sensitive nature of this data requires efficient, privacy-preserving, locally-deployed solutions. This paper compares the feasibility of various local NLP models for this task. We evaluated a feature-engineered Conditional Random Field (CRF), a fine-tuned spaCy NER model, and several Large Language Models (LLMs) (1.1B to 14B parameters) on a new synthetic dataset of 1,000 Dutch debt letters. Models were compared using accuracy (F1-score) and deployment metrics (CPU runtime, memory usage). Our results show a clear performance-resource trade-off. Lightweight CRF and spaCy models efficiently extracted structured data but failed in many critical unstructured fields. In contrast, LLM performance scaled directly with model size. The 14B DeepSeek model achieved the highest accuracy (95.2% average F1), successfully handling all field types. In conclusion, larger local LLMs are the most viable solution for accurate, private document processing. Alternatively, a hybrid approach using lightweight models for structured data and LLMs only for complex, unstructured fields, would also be adequate.

Keywords: Large Language Models (LLMs), Information Extraction, Named Entity Recognition (NER), Debt Collection Letters, OCR (Optical Character Recognition)

1. Introduction

Debt collection letters are important financial documents delivered to individuals and organizations to inform them of their outstanding financial obligations. In the Netherlands, millions of financial notices are issued annually by bodies like the Belastingdienst (National Tax Service) and the Centraal Justitieel Incassobureau (CJIB). This high volume of such notifications makes them a common part of daily life, with significant societal impact (Burgt et al., 2025).

Debt letters are frequently unstructured, contain legal jargon, and vary significantly in layout, as each creditor may use a different template. For recipients, this complexity can be intimidating, leading to misinterpretation, postponed payments, and deepening financial distress. For the institutions, municipalities, and legal advisers trying to help, the initial stage of any workflow (manually interpreting letters to extract key data like debtor names, reference numbers, and due dates) is time-consuming and error-prone.

Automating such information extraction is essential but technically difficult. The lack of layout standardization, combined with noise from Optical Character Recognition (OCR) on scanned documents, complicates automated processing. Furthermore, these letters contain highly sensitive personal data, raising privacy concerns. This constraint rules out powerful, cloud-based AI solutions and creates a pressing need for accurate, lightweight models that can be deployed locally. Developing a solution that is both accurate and privacy-preserving remains an open technical challenge (Truhn and Kather, 2024).

To address this challenge, we investigate the feasibility of using lightweight, locally-deployed Natural Language Processing (NLP) models to solve this extraction task. We compare the performance of traditional machine learning (Conditional Random Fields), modern neural architectures (spaCy NER), and Large Language Models (LLMs) on a synthetic corpus of noisy Dutch debt letters. Our evaluation extends beyond standard technical metrics (accuracy, F1-score, recall) to include practical concerns for deployment (runtime, memory usage).

Specifically, we aim to answer the following research questions:

- **RQ.1:** Which local Natural Language Processing model is most suitable for extracting structured data from Dutch debt collection letters, considering accuracy, robustness to noise, computational efficiency, and deployment feasibility?
- **RQ.2:** What are the trade-offs between these models in terms of accuracy, cost, inference time, and ease of integration into local document processing workflows?

Proceedings of the Fifteenth Language Resources and Evaluation Conference (LREC 2026), pages 10140–10149
11–16 May 2026.

2. Background and related work

Traditional approaches to information extraction rely on statistical sequence models like **Conditional Random Fields (CRFs)**. CRFs are effective at sequence labeling because they consider the context of neighboring tokens and labels when making predictions (Lafferty et al., 2001). In practice, this involves extensive domain-specific feature engineering, where researchers define explicit rules and patterns (such as token capitalization, the presence of digits, or proximity to specific keywords like "Invoice") to identify entities. While CRFs have proven successful and serve as a strong baseline in Named Entity Recognition (NER) challenges, their performance depends on the quality of these handcrafted features. This makes them difficult to adapt to the wide variability of layouts found in real-world documents like invoices, requiring costly and continuous expert intervention. Recent comparative studies have confirmed the limitations of CRFs in specialized domains, highlighting their sensitivity to dataset noise and the superior performance of transformer-based models (Pamio and Di Nunzio, 2025).

The rise of deep learning led to the adoption of pre-trained language models, such as BERT, and frameworks like **SpaCy NER** (Honnibal and Montani, 2021), which automated the feature engineering process. These models learn rich, context-sensitive word representations from vast amounts of unlabeled text, which can then be fine-tuned for specific NER tasks. For example, Hamdi et al. (2021) adapted BERT-based architectures for extracting key fields from invoices, demonstrating strong performance. A survey by Saout et al. (2024) documents the successful application of deep learning, including BERT and other neural architectures, to invoice data extraction. Similarly, the PBa-LLM framework leverages models like BERT and RoBERTa to identify sensitive entities in resumes for privacy-preserving NLP (Mancera et al., 2025). However, while these models eliminate the need for manual feature engineering, their main limitation is the requirement for large, domain-specific, and meticulously annotated datasets for fine-tuning. In many contexts, generating such datasets is impractical and costly, creating a significant bottleneck for deployment.

More recently, the paradigm has shifted toward **Large Language Models (LLMs)**, which can perform complex extraction tasks with minimal or no task-specific training through prompting (Agrawal et al., 2022). A comprehensive study by Ntinopoulos et al. (2025) evaluated 18 LLMs on data extraction from synthetic electronic health records and found that models like Claude 3.0 Opus and GPT-4 achieved outstanding accuracy (>98%) without any fine-tuning. Despite this power, deploying these models introduces a new set of challenges. The need to send sensitive information to third-party APIs presents a challenge for financial and legal applications. This has led researchers to use synthetic data to circumvent privacy issues, but it also highlights the need for local solutions. However, running capable open-source LLMs locally can be computationally intensive and require significant resources, posing another barrier for many organizations. This is a critical consideration, as recent work on resource-sensitive tasks notes that fine-tuned BERT-based models can remain competitive with, or even outperform, zero-shot LLMs while having significantly lower computational costs (Upravitelev et al., 2025). Although larger and larger LLMs have recently been released, the value of small LLMs has been recognized for localized, privacy sensitive, and resource-constrained settings (Belcak et al., 2025).

While these approaches (feature-engineered, fine-tuned, and prompted) have been explored individually, a direct comparative analysis under the constraints of local, privacy-preserving deployment for financial document processing remains a largely unaddressed research gap. This study aims to fill this gap by systematically comparing CRF, fine-tuned NER, and local LLM approaches on the task of extracting data from Dutch debt collection letters. The results provide an empirical baseline for managers who navigate decisions about privacy, precision, and resource feasibility.

3. Methods

We first explain how we created a synthetic dataset of Dutch debt letters, based on templates extracted from actual, non-synthetic debt letters. We then explain how we applied CRF, Spacy, and local LLMs to this dataset. Finally, we explain the evaluation procedure we applied to each model. Figure 1 provides an overview.

3.1. Synthetic Dataset Construction

Due to the sensitivity and limited availability of authentic Dutch debt collection letters, a synthetic dataset was constructed to closely mimic their content, structure, and visual appearance. The data generation process was supported using a custom Python script, enabling the creation of a large volume of diverse and realistic instances.

While the dataset is synthetic, its structure and phrasing were grounded in real Dutch debt collection letters to reflect typical layouts, tone, and commonly occurring fields. This is a key contribution of the work: it enables reproducible experimentation under privacy constraints while still approximating

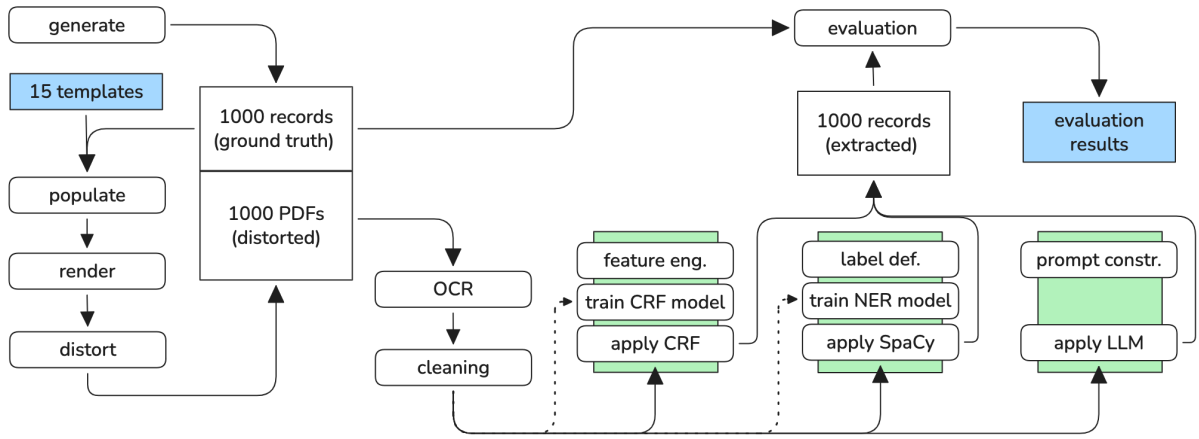

Figure 1: Overview of the experiment. The dataset of annotated synthetic debt letters is created on the left-hand side. After OCR and cleaning, the data is fed to three types of models to extract key data fields. An evaluation is performed to compare the extracted records against the ground truth. Data splits for training are indicated by dashed lines.

real-world variability, making the approach more readily transferable to authentic operational data.

1. Base templates and content insertion A set of letter templates was created to reflect the layout and tone commonly used by Dutch collection agencies. Each template included a header with the agency's address and a logo placeholder, a salutation, main body, and footer. Realistic values were inserted using the `Faker` library configured for the Dutch locale (`nl_NL`). This allowed for generation of debtor names and addresses, selection of creditor names from a predefined list, randomized assignment of plausible dates and monetary amounts, and insertion of legal boilerplate text from a curated library. This approach ensured that each letter maintained the formal tone and verbosity characteristics of official correspondence.

2. Formatting and rendering The `ReportLab` library was used to render the populated templates into styled PDF documents. Layout features such as type fonts, text alignment, spacing, and address block positioning were varied to simulate different letterhead styles. Additional elements such as reference numbers and QR codes were incorporated to enhance authenticity. This stage produced clean, high-resolution PDFs representing the digital form of debt letters.

3. Distortion and OCR text extraction To emulate real-world conditions in which debt letters are often received as scanned copies, data augmentation was applied to introduce controlled visual distortions while preserving semantic content. Transformations included down-sampling to lower resolutions, applying Gaussian noise and blur, introducing slight page rotation or skew, and simulating font degradation effects. The resulting images were processed using the Tesseract OCR engine to extract text content. The OCR output contained typical recognition errors, such as misreading numeric digits as letters or vice versa (e.g., the digit '2' as the letter 'Z' or the digit "0" as the letter "O") or introducing minor currency symbol misinterpretations (e.g., '€104.02' as 'e104.02'). These imperfections created realistic test conditions for assessing the robustness of NLP extraction methods.

4. Annotation generation As all inserted content was programmatically generated, ground truth labels for each field were retained. A structured JSON annotation was saved for every letter, mapping each field name to its corresponding value, including debtor details, creditor information, financial amounts, and dates. This standardized annotation format facilitated supervised training and precise evaluation by direct comparison of model predictions with ground truth values.

Final dataset structure Each synthetic data point consisted of a visually distorted PDF image and a corresponding JSON annotation. Approximately 1,000 letters were generated, with 800 allocated for training and 200 reserved for testing. Templates were shared across splits to evaluate generalization, and randomized content ensured that models were required to handle unseen values.

Preprocessing and cleaning Before the synthetic data was passed to each of the models, a common preprocessing and cleaning pipeline was applied. The extracted text was cleaned to reduce noise by normalizing common OCR typos, removing extra whitespace, fixing broken tokens using heuristics, and splitting paragraphs into sentences to improve downstream token-level model performance. This preprocessing ensures that the model input is as consistent and readable as possible.

3.2. CRF field extraction

A Conditional Random Field (CRF) model was developed to extract structured data from OCR-processed letters. This approach treats the text as a sequence of tokens, each labeled with a data field or "O" for tokens outside any entity. We use `sklearn-crfsuite` for training and inference.

1. Labeling and model configuration Tokens are annotated using the BIO (Begin, Inside, Outside) scheme, which supports multi-token spans such as full names or addresses. The CRF is configured with L-BFGS optimization and L1/L2 regularization to prevent overfitting, balancing convergence performance and generalization.

2. Feature engineering for tokens The CRF model's effectiveness relies on a rich feature set, designed from domain knowledge of debt letter structure. Lexical features include lowercase tokens, prefixes/suffixes, and token length. Character-type features indicate casing, numeric status, or the presence of special symbols such as the euro sign. Regex-based indicators detect patterns such as dates, euro amounts, IBANs, Dutch postal codes, phone numbers, invoice numbers, and email addresses. Dictionary features flag words related to amounts, dates, names, months, addresses, and company types. Context features capture patterns in surrounding tokens, which is particularly valuable in formal correspondence where honorifics and salutations precede names. For example, in the phrase "Geachte mevrouw Janssen," capitalization and position relative to preceding formal words strongly suggest that "Janssen" is a name. Positional features capture whether a token appears early or late in the document or at the start of a line, exploiting layout tendencies.

3. Training procedure The CRF is trained on tokenized OCR text paired with BIO labels. Tokenization uses NLTK's Dutch `word_tokenize` with custom logic to preserve critical punctuation. Features are generated for each token, and the model is trained on an 80/20 train-test split, providing robust evaluation while maintaining sufficient training data.

4. Inference and reconstruction During inference, tokens are processed into features and passed to the CRF to predict BIO labels. Contiguous B- and I-tags are merged to reconstruct entities such as names or addresses. Postprocessing could be applied to normalize fields like dates or IBANs, but we chose to use raw predicted spans for evaluation.

3.3. SpaCy NER field extraction

A Named Entity Recognition (NER) pipeline was developed using SpaCy to extract key data fields from OCR-processed debt collection letters. This pipeline identifies fields such as names, amounts, dates, and account numbers directly from noisy letter text. The aim is to produce structured JSON outputs that match the ground truth in the synthetic dataset. The pipeline is constructed as follows.

1. Label definitions and mapping The first step is defining a fixed set of entity labels corresponding to data fields annotated in the JSON files, such as `debtor_name`, `debt_amount`, and `letter_date`. These are mapped to SpaCy-compatible uppercase entity labels such as `DEBT_AMOUNT` or `CREDITOR_ACCOUNT`. Declaring labels explicitly and using them consistently throughout training ensures compatibility with SpaCy's internal mechanisms and alignment between annotations and model outputs.

2. Training data construction Training examples are created by loading the text and corresponding JSON annotations, locating spans for each field, and converting valid spans into `char_span()` objects in a SpaCy document. These are compiled into example objects for supervised NER training, prioritizing longer spans to avoid overlap with shorter substrings. This guarantees that each entity is correctly located, enabling the model to learn and generalize patterns effectively.

3. NER model training The Dutch language model `nl_core_news_lg` is used as a base and extended with a custom NER component. During training, examples are shuffled and passed to SpaCy's training loop, with weights updated via backpropagation. Fine-tuning a pre-trained model leverages existing linguistic features while adapting to the debt collection domain, which is more efficient and effective than training from scratch.

4. Inference and field recovery At inference time, a letter's OCR text is passed through the trained model, and detected entities are mapped to their original keys. If multiple entities of the same label occur, a selection strategy is applied; we simply chose the first occurrence. This process transforms noisy, unstructured text into a clean, structured data dictionary suitable for automated processing.

3.4. Local LLM field extraction

A third approach uses instruction-following local LLMs to interpret full letter text and extract structured data without token-level supervision. Three models were selected: TinyLLaMA (1.1B parameters), suitable for resource-constrained devices; Gemma (4B), offering balanced performance and compute efficiency; and DeepSeek (14B), representing higher-end local model capabilities. These models were chosen to represent different model sizes and resource requirements, from a lightweight

model (TinyLLaMA) to a more capable, high-performance model (DeepSeek). All were run offline as quantized `gguf` versions via `llama_cpp`, with an 8K context window and CPU-only inference.

1. Prompt engineering We designed a concise, domain-specific system prompt that directs the model to extract fifteen predefined metadata fields from OCR text and return them as a structured JSON object. Missing values must be labeled "None," and the model must not produce any extraneous text, ensuring consistent, machine-readable outputs across models. The complete prompt operationalizing this objective is shown below:

```
You are an AI assistant specialized in analyzing
debt collection letters. Extract and organize the
following information from the provided debt letter:

    1. creditor_account
    2. subject
    3. client_number
    4. letter_date
    5. creditor_email
    6. deadline_date
    7. invoice_number
    8. invoice_date
    9. letter_id
    10. debt_amount
    11. original_amount
    12. debtor_name
    13. debtor_address
    14. creditor_name
    15. creditor_address

Format your response as a structured JSON object
with the keys mentioned above. Your response
should contain 15 keys and values. No other output
is needed, return a single JSON object. If any
information is not found, indicate it as "None".
```

This prompt is wrapped in special tokens (`<system>`, `<user>`) and appended with the full OCR-extracted text of a synthetic debt letter. The model is then asked to generate a structured report as output.

2. Inference and output parsing The prompt, combined with the OCR text, is processed via the `create_completion()` method. Parameters are tuned for deterministic, structured output: `max_tokens=1024` allows for complete JSON generation, `temperature=0.1` minimizes randomness, and stop tokens ensure the output terminates cleanly.

3.5. Evaluation Procedure

All models produce a structured JSON output and are evaluated using a common scoring tool that compares predictions with ground truth using exact and fuzzy matching.

Evaluation dataset The evaluation set comprises 200 unseen debt letters, each with OCR-extracted text, ground truth annotations, and model predictions. Only fields present in at least half of the examples are evaluated to ensure statistical reliability.

Evaluation metrics Field-level metrics include exact match, partial match ($\geq$ 70% similarity using the `rapidfuzz` token sort ratio), incorrect/missing classification, and precision, recall, and F1-score. The 70% threshold balances tolerance for OCR errors against preventing false positives.

Field	P	R	F1
creditor_account	88.89%	31.37%	46.23%
creditor_email	100.00%	6.86%	12.83%
Average	**94.45%**	**19.12%**	**29.53%**
all other fields	0.00%	0.00%	0.00%

Table 1: Field-wise performance of the CRF model on test letters. Fields with 0.00% scores were either not predicted or failed evaluation due to limited training instances, ambiguous annotations, insufficiently distinctive token patterns, or inherent model limitations in capturing the relevant structure.

Evaluation protocol For each model, predictions are saved as JSON, evaluated per field, and averaged across all fields to obtain overall precision, recall, and F1-scores.

Hardware and runtime measurement All evaluations were run on a 2021 16-inch MacBook Pro with an Apple M1 Max chip and 32GB RAM, using macOS Sequoia 15.5. CPU-only inference was used for all models, and runtime per debt letter was measured using high-resolution timers and averaged over the 200 test samples. This ensures fair comparison of computational efficiency across approaches.

4. Results

This section presents the performance of the extraction pipelines on the synthetic debt letter dataset. All models were evaluated using the same dataset, metric definitions, and scoring script, ensuring direct comparability.

4.1. Baseline Model Performance

We first establish baseline performance using two traditional NLP approaches: a feature-engineered Conditional Random Field (CRF) and a fine-tuned SpaCy NER model.

Conditional Random Field (CRF) The CRF model was trained on the synthetic debt letters using hand-crafted features such as token casing, digit patterns, and surrounding context. The model only identified two out of 15 fields with minimal success (Table 1), bringing field coverage to only 13%. The model's low recall may suggest that the CRF's reliance on specific, rigid token-level patterns (e.g., capitalization, surrounding words) caused it to fail when these patterns were not perfectly consistent

Field	Precision	Recall	F1-score
creditor_account	97.96%	97.96%	97.96%
subject	91.84%	97.83%	94.74%
client_number	91.58%	97.75%	94.57%
letter_date	89.80%	97.78%	93.62%
creditor_email	88.54%	95.51%	91.89%
deadline_date	97.56%	86.02%	91.43%
invoice_number	80.41%	96.30%	87.64%
invoice_date	65.22%	89.55%	75.47%
letter_id	61.22%	96.77%	75.00%
Average	**84.46%**	**95.83%**	**89.50%**
all other fields	0.00%	0.00%	0.00%

Table 2: Field-wise performance of the SpaCy NER model on 200 test letters. Fields are sorted on F1-score. Averages are calculated only over fields with some successful extraction.

across the test documents. This highlights the limitations of feature-engineered approaches on semi-structured, noisy text.

SpaCy NER The SpaCy model, fine-tuned using annotated synthetic letters, achieved strong performance on several structured data fields. As shown in Table 2, it excelled at identifying entities with consistent formats, such as `client_number`. However, it failed entirely on several critical fields, indicating a difficulty in generalizing to less structured information, such as `debt_amount`. Since 9 out of 15 fields are extracted with some success, field coverage is 60%. When all fields are considered, including those with 0.00% scores, the SpaCy model achieved an average F1-score of 53.49%.

4.2. Large Language Models

We evaluated three LLMs of varying sizes. As shown in Table 3, LLM performance scales with parameter count. TinyLLaMA (1.1B), a compact model, struggled to produce accurate outputs, reflected in its very low F1-scores and field coverage of 40%. Gemma (4B) showed competence on structured fields (e.g., dates, emails), offering a balance of capability and size. DeepSeek (14B) showed the highest performance across nearly all categories, successfully extracting both clearly formatted fields and unstructured text (e.g., names, addresses) with high accuracy. Field coverage is 100% for both Gemma and DeepSeek.

4.3. Overall comparison and analysis

A direct comparison of the LLMs reveals clear trade-offs between model size, performance, and computational cost. Table 4 provides an overview.

Performance vs. Resource trade-offs The results show a clear hierarchy: DeepSeek (14B) achieved the highest overall performance with an average F1-score of 95.2%, reliably extracting both structured and less structured fields, albeit with higher runtime and memory usage. Gemma (4B) provided a moderate balance between accuracy (78.9% average F1) and resource requirements, excelling in fields that occur often while moderately handling complex entities.

The baseline models (CRF and SpaCy) and TinyLLaMA were significantly faster and lighter but were not reliable enough for this comprehensive extraction task. SpaCy NER performed well on structured entities like creditor accounts, client numbers, and letter dates, yet failed entirely on critical financial fields such as debt amounts and addresses. CRF achieved high precision on structured fields such as creditor accounts and emails but showed very low recall and poor generalization to diverse layouts. TinyLLaMA (1.1B) had the weakest performance (F1 = 15.9%), hindered by small capacity and inconsistent output formatting.

Overall, these results highlight a trade-off between interpretability and efficiency in classical models (CRF, SpaCy) versus flexibility and coverage in local LLMs (Gemma, DeepSeek). This represents a critical decision point for deployment: sacrificing peak accuracy for efficiency or investing more resources to achieve higher reliability.

Error analysis Qualitative inspection revealed model-specific error patterns. Notably, the deliberately introduced OCR artifacts did not appear to be the dominant source of failure; instead, the most frequent issues in the error analysis reflected model limitations in extraction consistency and structured output generation. TinyLLaMA often hallucinated values and produced invalid JSON. CRF relied too heavily on surface cues and missed multi-token spans. SpaCy struggled with financial fields that lacked distinct token-level patterns. Even the better-performing LLMs made errors; Gemma and, to a lesser extent, TinyLLaMA sometimes confused conceptually related fields such as `debt_amount` vs. `original_amount`, and `creditor_address` vs. `debtor_address`. Such confusions could propagate into downstream systems, highlighting the need for post-processing or disambiguation strategies. DeepSeek was the most robust but occasionally duplicated fields or even hallucinated extra fields in its JSON output.

5. Discussion

In this section, we synthesize the key findings to answer the research questions on model suitability and performance trade-offs. We then contextualize these technical results using domain-specific observations from field interviews.

Field	TinyLLaMA			Gemma			DeepSeek		
	P	R	F1	P	R	F1	P	R	F1
creditor_name	0.0%	0.0%	0.0%	50.0%	50.0%	50.0%	98.5%	99.0%	98.8%
creditor_email	75.0%	30.0%	42.9%	100.0%	99.0%	99.5%	99.0%	98.0%	98.5%
debtor_name	100.0%	9.1%	16.7%	50.0%	40.0%	44.4%	98.0%	99.0%	98.5%
creditor_address	0.0%	0.0%	0.0%	50.0%	55.0%	52.4%	97.5%	98.0%	97.8%
debtor_address	0.0%	0.0%	0.0%	60.0%	55.0%	57.4%	97.5%	97.5%	97.5%
invoice_number	0.0%	0.0%	0.0%	77.0%	85.0%	80.8%	98.0%	96.0%	97.0%
invoice_date	0.0%	0.0%	0.0%	65.0%	78.0%	70.9%	97.0%	97.0%	97.0%
debt_amount	0.0%	0.0%	0.0%	60.0%	50.0%	54.6%	97.0%	96.0%	96.5%
subject	0.0%	0.0%	0.0%	88.0%	91.0%	89.5%	96.0%	97.0%	96.5%
original_amount	0.0%	0.0%	0.0%	55.0%	48.0%	51.3%	95.0%	97.0%	96.0%
deadline_date	0.0%	0.0%	0.0%	89.0%	79.0%	83.7%	95.0%	93.0%	94.0%
letter_date	25.0%	12.5%	16.7%	94.6%	94.6%	94.6%	86.5%	95.4%	90.7%
client_number	33.3%	28.6%	30.8%	93.8%	97.8%	95.7%	92.9%	84.8%	88.6%
letter_id	33.3%	28.6%	30.8%	89.6%	46.7%	61.4%	92.1%	76.1%	83.3%
creditor_account	20.0%	14.3%	16.7%	73.5%	91.0%	81.3%	100.0%	83.3%	91.2%
Average	**26.5%**	**11.6%**	**15.9%**	**76.3%**	**82.2%**	**78.9%**	**96.8%**	**93.7%**	**95.2%**

Table 3: Field-wise performance comparison of TinyLLaMA, Gemma, and DeepSeek models (sorted by DeepSeek F1 scores, percentages rounded to one decimal place).

Model	#P(B)	P(%)	R(%)	F1(%)	s	MB
TinyLLaMA	1.1	26.5	11.6	15.9	2.1	320
Gemma	4.0	76.3	82.2	78.9	4.7	7000
DeepSeek	14.0	96.8	93.7	95.2	6.3	9000

Table 4: Model-level comparison of performance, size, runtime, and memory usage. where s is average seconds per letter on CPU and MB is approximate peak resident memory during inference.

5.1. Key findings

This study's findings provide answers to the central research questions by evaluating model performance across accuracy, robustness, computational efficiency, and deployment feasibility.

Our analysis of model trade-offs (RQ2), synthesized in Table 5, reveals a clear performance-versus-resource pattern. Traditional models (CRF, SpaCy NER) are efficient but lack the flexibility for diverse layouts, with critical failures on specific fields limiting their standalone utility. Conversely, local LLM performance scales directly with size: smaller models like Gemma (4B) offer a moderate balance of accuracy and computational demand, while DeepSeek (14B) delivers robustness at a significantly higher resource cost.

Regarding RQ1, the most suitable local model depends on specific deployment priorities. For maximum accuracy (e.g., in high-stakes compliance workflows), DeepSeek (14B) is the most suitable model, justifying its high computational cost. While computationally efficient, the CRF, SpaCy NER and Gemma (4B) models proved unsuitable as standalone solutions due to their critical failures when extracting unstructured fields. These models could be useful in a hybrid solution where more resource-heavy LLMs are only used for fields where these more efficient models do not perform at an acceptable level. Finally, TinyLLaMA (1.1B), despite its lightweight nature, is unsuitable for practical deployment due to its unreliable outputs.

5.2. Generalizability

We note that the proposed pipelines can be generalized to other languages and document genres with modest adaptations. In addition, several practical improvements could increase robustness when generalizing to new creditors, templates, and document conditions. For CRFs, performance gains typically require iterative, manual feature and lexicon updates as new formats emerge, which can be costly to maintain at scale. SpaCy could benefit from additional domain-specific training data and targeted rule support (e.g., phrase-matching and lightweight pattern-based preprocessing) to better capture sparse financial fields. For LLM-based extractors, reliability can be strengthened by adding deterministic verification steps (schema validation, type checks, value-range constraints) and, where feasible, constrained decoding to enforce JSON compliance. Finally, hybrid strategies that combine structured models (CRF/SpaCy or rule-based components) with LLM flexibility offer a promising path forward, especially for disambiguating conceptually similar fields while keeping latency and computational overhead manageable.

For the CRF, language and domain specific features (token shape, currency symbols, postal codes, honorifics, and date/currency conventions)

Model	Advantages	Disadvantages
CRF	Lightweight, interpretable, precise for structured fields	Limited generalization, requires manual feature engineering, poor recall on diverse layouts
SpaCy NER	Context-aware, privacy-friendly local deployment, good for structured entities	Needs extensive labeled data, struggles with ambiguous or variably structured fields
TinyLLaMA (1.1B)	Very lightweight, fast inference, minimal hardware requirements	Low accuracy, unstable outputs, unreliable for critical data extraction tasks
Gemma (4B)	Balanced accuracy and efficiency, suitable for moderate-resource environments, consistent across many fields	Still computationally heavier than classical models, does show instability in edge cases
DeepSeek (14B)	High accuracy; robust generalization, zero-shot capability, handles structured and unstructured fields well	High computational cost, deployment complexity in low-resource settings

Table 5: Comparison of evaluated models for data extraction from debt letters

can be respecified, while retaining layout agnostic positional cues to mitigate format drift. For SpaCy, portability is achieved by swapping in a language appropriate base model, aligning the label schema, and fine-tuning on a small, stratified sample of target-language letters; weak supervision (e.g., pattern/regex heuristics for dates, amounts, IBAN formats) can further reduce the annotation burden. For LLMs, the extraction prompt can be translated and augmented with locally specific field definitions and canonical output formats (e.g., ISO-8601 dates, normalized currencies), and constrained decoding can stabilize JSON compliance across domains.

In all cases, robustness to domain shift (new creditors, templates, or scan qualities) benefits from (i) augmenting OCR noise to match target conditions, (ii) validating with a language and domain-balanced development set, and (iii) conducting error taxonomies to identify fields requiring rule-based post-processing. Finally, the evaluation protocol itself is portable: exact/partial-match criteria, per-field precision, recall, and F!, and runtime/memory profiling can be reused unchanged, enabling comparisons when extending to other correspondence (e.g., utility bills, tax notices, bank statements, medical records) or additional languages. Notably, most of such administrative correspondence has highly regular, template-like layouts (fixed headers, repeated phrasing, and consistent field locations), which in principle should make it easier to process than more free-form documents; however, real-world variation across some fields and the degradation introduced by scanning/OCR can still erode these structural advantages.

5.3. Connection to field insights

In addition to the technical study, qualitative insights were obtained through interviews with a debt counselor from a local municipality and a representative of a private debt collection organization. The selected stakeholders offer two perspectives: the advisory function assisting individuals in financial trouble and the operational aspect of dispatching and managing debt correspondence.

The debt advisor mentioned that numerous people seeking assistance are overwhelmed by the correspondence they receive. Specifically, they find it challenging to figure out the subject of the letter, the amount owed, and the necessary steps to take. The adviser observed that numerous clients arrive with bags containing unopened or half-read correspondence, frequently attributable to feelings of guilt, stress, or a lack of understanding regarding the material. The counselor indicates that the most essential information includes the total debt amount, payment date, and contact information of the issuing agency; nevertheless, these elements are frequently obscured by convoluted legal terminology or inconsistent formats.

The debt collection agency representative affirmed that their templates differ based on the creditor they serve, and the language is deliberately formal to guarantee legal enforcement. They recognized that this results in letters being challenging for readers to comprehend. The agency indicated that their existing workflow necessitates considerable human effort to identify essential fields during customer responses or case transfers. They indicated interest in automation solutions; nevertheless, they acknowledged that privacy and document diversity present significant obstacles.

The interviewees highlighted the significance of obtaining structured data from debt letters. They emphasized the realistic limitations and requirements of both public assistance services and private collection organizations, which informed the evaluation criteria used in this study.

6. Conclusion

This study presented a comparative evaluation of three approaches to data extraction from Dutch debt collection letters under privacy-preserving,

local deployment constraints. Using a synthetic dataset augmented with OCR noise, we examined a feature-engineered Conditional Random Field, a fine-tuned SpaCy NER model, and instruction-following local LLMs. The results show that while CRFs and SpaCy remain competitive for structured fields, they lack the flexibility to handle diverse and noisy real-world documents. Local LLMs, particularly larger models such as DeepSeek 14B, achieved the highest accuracy and robustness across fields, though at the cost of higher runtime and memory demands. These findings indicate a clear trade-off between efficiency and coverage: classical models are lightweight, whereas local LLMs offer broader extraction capability but require greater computational resources.

This trade-off highlights the need to tailor model selection to specific operational constraints and accuracy requirements. Practical deployment requires balancing computational resource availability, privacy requirements, and accuracy criticality. For scenarios prioritizing accuracy, such as financial compliance, DeepSeek justifies the resource investment. Gemma serves adequately in contexts demanding moderate accuracy with constrained resources. While CRF and SpaCy are computationally efficient and privacy-friendly, their inability to consistently handle complex, high-risk extraction tasks limits their use as standalone solutions.

In conclusion, the main contribution of this work is a novel empirical comparison that quantifies this performance-versus-resource trade-off. This comparative analysis provides a practical guide for organizations seeking to implement automated document processing solutions that respect data privacy and align with their operational capabilities.

7. Limitations and future work

This research acknowledges several limitations. Despite these constraints, the controlled setup offers valuable insights into model behavior under noisy, privacy-preserving conditions.

Firstly, the synthetic dataset, while carefully constructed, may not capture all structural and linguistic edge cases found in authentic debt collection letters. Real-world documents often contain more diverse formatting styles, ambiguous phrasing, and OCR noise than those simulated here. At the same time, the use of synthetic data should be viewed as a limitation primarily from an evaluation standpoint, rather than as a barrier to applicability. Our synthetic letters were derived from real debt letters and designed to mirror their layout conventions, phrasing, and field structure as closely as possible, making them highly realistic proxies for the target domain. As such, this study functions as a proof-of-concept under privacy-preserving conditions, and the same design choices (noise-aware preprocessing, field-level evaluation, and error-driven post-processing) are actively being transferred to real-world settings. While we cannot report case-level outcomes or share representative examples here due to privacy and legal constraints, ongoing application to authentic debt letters provides practical evidence that the proposed pipeline is viable beyond the synthetic benchmark.

In particular, the OCR perturbations introduced in our synthetic pipeline may be too controlled and "balanced" compared to real-world conditions; modern OCR errors in practice can be messier and more uneven, which may reduce downstream extraction performance beyond what is observed in our experiments. That said, modern OCR systems have improved substantially in robustness and accuracy, so OCR quality is unlikely to be the primary bottleneck for either the study's conclusions or real-world deployment; instead, remaining errors can typically be handled through targeted preprocessing and post-processing.

Secondly, the evaluation assumes one correct value per field, potentially overlooking cases where multiple valid entries exist. Stringent matching criteria may have penalized near-correct results, while annotation variability could have affected reliability. LLM results are also sensitive to prompt phrasing and runtime conditions, leading to inconsistent outputs that complicate deterministic evaluation.

Future research should prioritize enhancing SpaCy's financial entity extraction capabilities, exploring hybrid models that combine structured and flexible approaches, optimizing LLM deployment, and validating models against authentic debt letters under operational conditions. Additionally, exploring deterministic constraints and ensemble methods for LLMs to address output variability would be beneficial. Future studies should extend the evaluation to different languages and a broader range of document types beyond debt collection letters to further test portability and robustness. As these extensions will require adapting language- and domain-specific preprocessing and post-processing (e.g., OCR normalization rules, format heuristics, and field definitions), the comparative framework and insights from this study should still transfer and inform future experiments under the same evaluation protocol.

We have started to operationalize several of these insights in a real life project, where we adopt a hybrid extraction strategy for real debt letters. Concretely, we combine lightweight, rule-based components for highly regular fields (e.g., fixed-format dates, currency amounts, IBAN-like patterns, recurring header blocks, and creditor reference formats) with an LLM-based extractor for less structured or more context-dependent fields (e.g., payment

conditions, action requirements, and letter-specific nuances). The rule-based layer serves both as a precision-oriented extractor for predictable fields and as a validator that constrains or cross-checks model outputs, while the LLM provides flexibility under template drift and diverse phrasing. In a real-world pipeline, this hybrid approach is embedded in an end-to-end flow: letters are ingested, OCR and normalization are applied, candidate fields are detected via deterministic heuristics, and the remaining information is extracted and explained conversationally by the LLM, with structured outputs written back into the system for downstream casework. This hybrid design directly follows the study's central takeaway: stable, high-precision extraction is best achieved by combining deterministic signals for regular fields with flexible language understanding for ambiguous or variable content.

8. Bibliographical References

Monica Agrawal, Stefan Hegselmann, Hunter Lang, Yoon Kim, and David Sontag. 2022. Large language models are few-shot clinical information extractors. In *Proceedings of the 2022 conference on empirical methods in natural language processing*, pages 1998–2022.

Peter Belcak, Greg Heinrich, Shizhe Diao, Yonggan Fu, Xin Dong, Saurav Muralidharan, Yingyan Celine Lin, and Pavlo Molchanov. 2025. Small language models are the future of agentic AI. arXiv:2506.02153.

Henry van der Burgt, Joost Beuving, Maurice Gesthuizen, and Toon van Meijl. 2025. Class and kinship in problem debt: Navigating the "debt maelstrom" in the netherlands. *Focaal*, 2025(101):25–38.

Ahmed Hamdi, Elodie Carel, Aurélie Joseph, Mickael Coustaty, and Antoine Doucet. 2021. Information extraction from invoices. In *International Conference on Document Analysis and Recognition*, pages 699–714. Springer.

Matthew Honnibal and Ines Montani. 2021. SpaCy: Industrial-strength natural language processing in python. https://spacy.io. Accessed: 2025-06-01.

John Lafferty, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning (ICML)*, pages 282–289.

Gonzalo Mancera, Aythami Morales, Julian Fierrez, Ruben Tolosana, Alejandro Peña, Miguel Lopez-Duran, Francisco Jurado, and Alvaro Ortigosa. 2025. Pba-llm: Privacy-and bias-aware nlp using named-entity recognition (NER). In *International Conference on Document Analysis and Recognition*, pages 3–20. Springer.

Vasileios Ntinopoulos, Hector Rodriguez Cetina Biefer, Igor Tudorache, Nestoras Papadopoulos, Dragan Odavic, Petar Risteski, Achim Haeussler, and Omer Dzemali. 2025. Large language models for data extraction from unstructured and semi-structured electronic health records: a multiple model performance evaluation. *BMJ health & care informatics*, 32(1):e101139.

Lorenzo Pamio and Giorgio Maria Di Nunzio. 2025. Comparing CRF vs BERT models for named entity recognition and relation extraction. In *Working Notes of CLEF 2025 - Conference and Labs of the Evaluation Forum (CLEF 2025)*, volume 4038 of *CEUR Workshop Proceedings*. CEUR-WS.org.

Thomas Saout, Frédéric Lardeux, and Frédéric Saubion. 2024. An overview of data extraction from invoices. *IEEE Access*, 12:19872–19886.

Daniel Truhn and Jakob Nikolas Kather. 2024. Privacy-preserving large language models for structured medical information retrieval. *npj Digital Medicine*, 7(1):1–9.

Max Upravitelev, Nicolau Duran-Silva, Christian Woerle, Giuseppe Guarino, Salar Mohtaj, Jing Yang, Veronika Solopova, and Vera Schmitt. 2025. Comparing LLMs and BERT-based classifiers for resource-sensitive claim verification in social media. In *Proceedings of the Fifth Workshop on Scholarly Document Processing (SDP 2025)*, pages 281–287. Association for Computational Linguistics.