PM3-KIE: A Probabilistic Multi-Task Meta-Model for Document Key Information Extraction

Anonymous ACL submission

Abstract

Key Information Extraction (KIE) from visually rich documents (VRDs) is typically framed as either fine-grained token classification or coarse-grained entity retrieval. Token-level models effectively capture spatial and visual information associated with document spans, while entity-level models excel in modeling logical dependencies and align more closely with real-world use cases. This work introduces PM3-KIE, a probabilistic, multi-task meta-model that integrates fine-grained and coarse-grained approaches, leveraging the strengths of both paradigms. The proposed model introduces two key innovations: domain-specific schema constraints to enforce logical consistency and mitigate extraction errors, and the integration of large language models (LLMs) to validate extractions through semantic plausibility. Experimental evaluation on the public VRDU dataset demonstrates that PM3-KIE significantly outperforms three state-of-the-art models and a stacked ensemble, achieving a 2.5% improvement in F1 score, highlighting the model's efficacy in unifying fine- and coarse-grained representations for enhanced KIE performance.

1 Introduction

KIE is a crucial task in automating business document processing, with applications spanning finance, legal, and supply chain management. KIE automation is essential for reducing operational costs; for example, processing a single invoice can cost $13,11 and take eight days (Girsch-Bock, Mary, 2024; Cohen and York, 2020). Despite recent advancements, KIE remains challenging, particularly for documents with complex schemas or semi-structured layouts, where state-of-the-art approaches often fall short (Wang et al., 2023c).

KIE aims to extract structured key-value pairs from VRDs (Huang et al., 2019) and is typically performed using one of two paradigms, displayed in Figure 1:

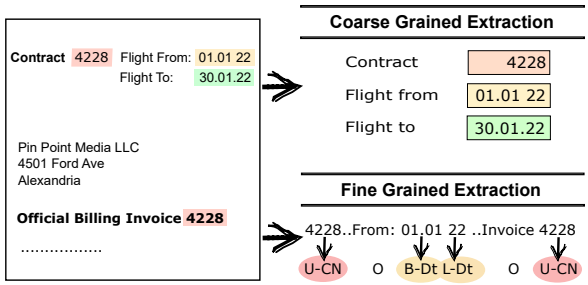

Figure 1: Fine-grained Token Classification vs. Coarse Grained Entity Extraction Task for KIE

Fine-grained token classification Models label individual tokens via sequence tagging, typically leveraging multi-modal transformer encoders that integrate text, spatial layout, and image features, such as LayoutLMv3 (Huang et al., 2022) and LiLT (Wang et al., 2022). While effective at capturing token-level spatial information, these models require extensive annotated data.

Coarse-grained entity extraction Models either generate structured output in the form of key-value pairs o json (Cesista et al., 2024), retrieve individual entity values (Cao et al., 2023) or classify a predefined set of entity candidates (Majumder et al., 2020). These methods perform well in capturing logical dependencies and producing coherent, structured outputs for coarse-grained tasks.

Token-level models excel at capturing spatial relationships, while entity-level models better capture logical dependencies, highlighting the need for approaches that integrate both. Ding et al. (2024) bridge these paradigms and suggest a knowledge distillation model to acquire knowledge from both model types, but their model is limited to specific architectures and necessitates classification logits. Thus, an integration of generative decoder-based

approaches or large-scale LLMs producing direct structured output is not possible.

Moreover, existing methods neglect domain-specific constraints, resulting in schema violations such as failing to extract required elements or incorrectly including optional ones. While heuristic postprocessing can partially address these issues, it often falls short of ensuring logical consistency.

Common errors also include syntactic or semantic inconsistencies, such as incomplete addresses or implausible entity values that could easily be prevented with contextual validation.

To address these limitations, we propose **PM3-KIE**, a novel probabilistic multi-task meta-model designed for KIE. PM3-KIE employs a lightweight probabilistic reasoning layer, leveraging the Probabilistic Soft Logic (PSL) framework (Bach et al., 2017) to combine logical constraints with probabilistic inference. This design ensures robust reasoning across granularities while maintaining computational efficiency.

We evaluate PM3-KIE on the public VRDU dataset (Wang et al., 2023c), demonstrating significant improvements over state-of-the-art models and a stacked ensemble meta-model across various in-distribution, out-of-distribution, and low-resource scenarios. Our contribution are the following:

- We introduce a probabilistic meta-model that integrates fine-grained and coarse-grained models with key innovations:

  fine-grained and coarse-grained blackbox model integration: PM3-KIE is a meta-model that supports incorporation of arbitrary fine-grained and coarse-grained blackbox KIE models including generative decoder approaches producing structured output.

  Schema consistency: Domain-specific constraints ensure logical adherence to extraction schemas and contained required and optional elements, reducing schema-related errors.

  LLM-as a judge based validation: Pretrained LLMs as a "judge" validate extractions based on syntactic and semantic plausibility. This mechanism provides an additional layer of quality control, enabling the model to prevent errors such as incorrect formatting or implausible entity values.

- We achieve state-of-the-art performance on the VRDU dataset, outperforming baseline models by a significant margin.

The remainder of this paper is structured as follows: Section 2 reviews related work. Section 3 provides background on the KIE problem descriptions. Section 4 provides details about the probabilistic model architecture. Section 5 presents experimental results, and Section 6 concludes with a discussion and future directions.

2 Related Work

KIE involves identifying key-value pairs from documents for structured data extraction (Huang et al., 2019) and can be performed as fine-grained token sequence classification or coarse-grained entity extraction at the document level, detailed in the following subsections.

2.1 Fine Grained Token Classification Models

Traditional Models To overcome the limitations of template-based approaches (Chiticariu et al., 2013; Schuster et al., 2013), neural networks were introduced, defining KIE as a token sequence classification problem. Recurrent Neural Networks (RNNs) were first used to process floating text and classify tokens (Palm et al., 2017), with some models incorporating spatial features to better represent document layouts (Sage et al., 2019).

Subsequently, convolutional architectures were employed to capture a non-sequential spatial context (Yang et al., 2017; Borges Oliveira and Viana, 2017; Zhao et al., 2019; Katti et al., 2018; Denk and Reisswig, 2019; Zhang et al., 2020).

Graph-based models enhanced this by modeling the document as a graph to represent textual and spatial relationships. Graph Convolutional Neural Networks have been used to integrate textual, spatial, and visual information for KIE tasks (Yao et al., 2024; Shi et al., 2023; Lee et al., 2022, 2021; Wei et al., 2020; Yu et al., 2020; Hwang et al., 2020; Liu et al., 2019; Qian et al., 2019).

Multimodal Transformer Approaches With the rise of transformer-based architectures, encoder-based models have become a dominant approach for multi-modal token classification in KIE. These models have been enhanced in various ways to address the challenges of integrating textual and visual features in business documents. DocFormer (Appalaraju et al., 2021), LAMBERT (Garncarek et al., 2021), FormNet (Lee et al., 2022), and ERNIE-Layout (Peng et al., 2022) focus on capturing spatial relationships through attention mechanisms. LayoutLMv3 (Huang et al., 2022), LAM-

BERT, Docformer, Structext (Li et al., 2021), and UDOP (Tang et al., 2023) incorporate layout-aware embeddings to better represent document structures, a technique employed in models. Furthermore, LayoutMask (Tu et al., 2023), StructextV2 (Yu et al., 2023), Structext, UDOP, DocFormer, ERNIE-Layout and Wokung-Reader (Bai et al., 2023) have introduced pre-training tasks designed to leverage multimodal information. In LiLT, Wang et al. (2022) adapted the transformer concepts to work in a language-independent way.

Low Ressource Approaches Chen et al. (2023) propose a task-aware meta-learning framework with a hierarchical decoder and contrastive learning for out-of-distribution, few-shot entity extraction. QueryForm (Wang et al., 2023b) utilize a dual prompting mechanisms to integrate schema and entity queries for zero-shot entity extraction.

2.2 Coarse-Grained Entity Extraction

Traditional and Encoder-Decoder Architectures Other works frame KIE as an entity extraction task. This can involve detecting candidates and classifying them, as proposed in RELIE (Majumder et al., 2020), or directly generating structured outputs using encoder-decoder models such as TILT (Powalski et al., 2021) and Donut (Kim et al., 2022), with TILT being one of the first to utilize a decoder to generate sequences that contain all entities.

Generative LLMs Recent works leverage LLMs with natural language prompts for KIE using two primary strategies: **Per-Key Prompting**, where individual prompts query values for each key (e.g., DocLLM (Wang et al., 2023a), LayoutLLM (Luo et al., 2024)); and **Unified Prompting**, where a single prompt queries all keys simultaneously, producing either *plain text* outputs with all values (e.g., GenKIE (Cao et al., 2023)) or *structured outputs* such as key-value pairs or JSON (e.g., RASG (Cesista et al., 2024), ICL-D3IE (He et al., 2023), LMDX (Perot et al., 2024)).

GenKiE employ an encoder-decoder framework to integrate such prompts, where the encoder processes multimodal document data and prompts, while the decoder outputs plain text containing entity values. LayoutLLM and DocLLM employ layout-aware LLM pretraining and VRDU task specific fine-tuning, addressing KIE by querying individual values for each document key.

Prompting Strategies and Paradigms Recent works like RASG, ICL-D3IE and LMDX utilize existing LLMs and propose techniques for in-context learning, document layout encoding, and prompting for KIE. RASG introduces retrieval-augmented structured generation, enabling LLMs to generate schema-constrained outputs by encoding document layout as text and modeling structured output prediction as a tool-use approach. ICL-D3IE employs in-context learning with demonstration examples that highlight positional relationships and enforce structured output. LMDX is a layout encoding technique to incorporate document text and spatial layout into the prompt and a decoding mechanism to parse the extracted entities from the LLM output.

Joint Token Classification and Entity Extraction Models Ding et al. (2024) address both token classification and entity extraction tasks simultaneously with a student-teacher framework for knowledge distillation. However, this method is limited to architectures that generate classification logits, rendering it incompatible with e.g. decoder-based LLMs that produce structured output.

We propose a more flexible multi-task meta-learning approach that combines fine-grained and coarse-grained models, including black-box approaches. By enforcing consistency across tasks and utilizing domain-specific schemas, our method overcomes the limitations of existing systems.

3 KIE Problem Statement

3.1 Traditional KIE Learning Tasks

KIE is a core task in document information extraction, aiming to extract key-value pairs that align with a predefined schema (Huang et al., 2019). Traditional approaches address KIE as either:

Token Sequence Classification Task The goal is to assign a label $l_s \in T$ to each token w_s in a document d, where $w = \{w_s\}_{s=1}^{|S|}$ is the sequence of tokens, $|S|$ is the length of the sequence, and $T = \{t_i\}_{i=1}^{|T|}$ is the set of possible label types. The label set T includes a special "OTHER" label, indicating that the token does not belong to any predefined label. The model predicts the label sequence $\mathbf{l} = \{l_1, l_2, \ldots, l_{|S|}\}$.

Key-Value Pair (KVP) Prediction Task The task is to extract a set of key-value pairs $K = \{(k_n, v_n)\}_{n=1}^{|N|}$, where $k_n \in T$ is the field type, v_n is the associated field value, and $|N|$ is the number

of extracted pairs. Each v_n is typically a subsequence of the token sequence $w = \{w_s\}_{s=1}^{|S|}$.

3.2 Unified Prediction

We introduce a new formulation for the joint fine-grained and coarse grained extraction task.

Business documents often adhere to a schema specifying the structured knowledge to be extracted. A schema $T = \{t_i\}_{i=1}^{|T|}$ defines field types, such as *invoice number*, and is populated by extracting corresponding information from each document d.

We define *Fields* as concrete instances of a field type t_i. *Field Mentions* represent text spans in d referring to fields, where multiple mentions may correspond to the same field.

Token sequence classification predicts field mentions in d, while entity extraction predicts higher-level field instances. Our joint formulation integrates them to predict both field instances and their mentions, leveraging outputs from both tasks.

Meta-Model for Finegrained and Coarsegrained Models Let $O = \{O^q\}_{q=1}^{|Q|}$ denote a set of |Q| finegrained prediction models all producing a predicted label sequence $L^q = \{l_s^q\}_{s=1}^{|S|}$. Let $U = \{U^r\}_{r=1}^{|R|}$ denote a set of |R| coarse-grained prediction models all producing a set of key value pairs $K^r = \{(k_n^r, v_n^r)\}_{n=1}^{|N|}$ Given a document d, token sequence w and the output of prediction models O and U, we define two sets of candidates:

- **Field Mention Candidates** (M^i): For a given field type t_i, the mention candidate set is defined as a subset of tokens in w predicted to belong to label t_i by at least one model in O:

$$M^i = \{w_j \mid j \in \{1, \ldots, |S|\}, \exists\, q \in \{1, \ldots, |Q|\} l_j^q = t_i\}. \quad (1)$$

- **Field Candidates** (F^i): For a given field type t_i, the field candidate set is defined as a subset of all field values v_n predicted to belong to the type t_i by at least one model in U:

$$F^i = \{v_c \mid c \in \{1, \ldots, |N|\}, \exists\, r \in \{1, \ldots, |R|\}, k_c^r = t_i\}. \quad (2)$$

Objective is to predict field mentions, fields and links, represented by random variable x, y and a:

Field Mentions: $x_{ji} = \begin{cases} 1 & \text{if } w_j i \text{ is mention of type } t_i, \\ 0 & \text{otherwise.} \end{cases}$

Fields: $y_{ci} = \begin{cases} 1 & \text{if } v_c i \text{ is a field instance of type } t_i, \\ 0 & \text{otherwise.} \end{cases}$

Field Linking: $a_{jci} = \begin{cases} 1 & \text{if } w_j i \text{ is linked to } v_c i, \\ 0 & \text{otherwise.} \end{cases}$

The joint approach leverages task interdependencies both from mentions to fields and from fields to mentions: Mentions predicted in the token sequence classification task refine field extraction, while field extractions validate and enhance mention predictions, ensuring schema consistency.

4 Methodology

4.1 P3M-KIE overall model architecture

We propose PM3-KIE, a probabilistic multi-task meta-model for KIE, that integrates fine-grained and coarse-grained models to reason across granularities. The framework features a logical decision layer based on PSL (Bach et al., 2017), combining first-order logic with probabilistic graphical modeling, as detailed in Subsection 4.1. Subsequent sections elaborate on PM3-KIE's architecture . Subsection 4.3 presents the integration of fine- and coarse-grained models for joint predictions of field mentions and values. Subsection 4.4 introduces the modeling of field cardinalities, while Subsection 4.5 describes the incorporation of LLMs as semantic and syntactic validators. Finally, Subsection 4.6 outlines the learning and inference mechanisms within the proposed framework.

4.2 Probabilistic Framework

Probabilistic Graphical Model PSL represents a probability distribution over a set of random variables using a *Hinge-Loss Markov Random Field* (HL-MRF). The probability density function for unobserved variables $Y = (Y_1, \ldots, Y_{n'})$, conditioned on observed variables $X = (X_1, \ldots, X_n)$, is expressed as:

$$P(Y|X) = \frac{1}{Z(\omega, X)} \exp\left[-\sum_{j=1}^{m} \omega_j \phi_j(X, Y)\right], \quad (3)$$

where ϕ_j are potential functions, ω_j are their associated weights, and $Z(\omega, X)$ is the normalization factor. This formulation allows joint reasoning over interdependent variables.

Declarative Logic Rules PSL employs a declarative language to define logical rules as templates for potential functions in the HL-MRF. Each rule

consists of *predicates* representing observed or unobserved variables and *constants* serving as placeholders. For example:

$$w : \texttt{Prediction(}\textbf{\texttt{mention}}\texttt{, }\textbf{\texttt{type}}\texttt{)} \implies \texttt{IsType(}\textbf{\texttt{mention}}\texttt{, }\textbf{\texttt{type}}\texttt{).} \quad \text{(Rule 1)}$$

w indicates the rule's importance. Grounding replaces variables in the rule (e.g. `mention`, `type`) with constants, generating multiple ground rules.

Learning and Inference PSL learns optimal rule weights w by maximizing a likelihood function. Inference determines the most probable assignments for unobserved variables, framed as a convex optimization problem (see Bach et al. (2017) for a detailed description).

4.3 Multi-Task Metamodel Design

Integration of Fine-Grained Models We introduce an unobserved predicate $Me^i(\mathbf{d}, \mathbf{m})$ for each field type $t_i \in T$, where $\mathbf{d}$ represents a document, and $\mathbf{m}$ is a mention candidate in M^i. This predicate encodes the true field mention prediction for each candidate x_{ji}.

Let o^q_{ji} denote the output of a prediction model O^q for a specific label t_i and a mention candidate w_j in M^i with $j \in \{1, 2, \ldots, |S|\}$, and $q \in \{1, 2, \ldots, |Q|\}, l^q_j = t_i\}$. For models that predict a label for each word, we define $o^q_{ji} = \mathbf{1}\left(l^q_j = t_i\right)$, where $\mathbf{1}(\cdot)$ is an indicator function returning 1 if the label assigned to w_j by O^q is t_i, and 0 otherwise.

For models outputting a probability distribution over labels $t_i \in T$, o^q_{ji} is defined as the probability assigned by O^q to word w_j and label t_i: $o^q_{ji} = P(t_i \mid w_j, O^q)$,

We define an observed predicate $OM^{q,i}(\mathbf{d}, \mathbf{m})$ for each model O^q, representing the output o^q_{ji} for field type t_i. Here, $\mathbf{d}$ corresponds to the document and $\mathbf{m}$ to the mention candidate in M^i. This predicate encapsulates the model's predictions for each mention in a document.

Rule 2 and Rule 3 establish the relationship between the outputs of models O and the unobserved field mention predictions $Me^i(d, m)$ for each field type t_i and model O^q:

$$w^{li}_1 : OM^{q,i}(\mathbf{d}, \mathbf{m}) \Rightarrow Me^i(\mathbf{d}, \mathbf{m}), \quad \text{(Rule 2)}$$

$$w^{li}_2 : \neg OM^{q,i}(\mathbf{d}, \mathbf{m}) \Rightarrow \neg Me^i(\mathbf{d}, \mathbf{m}). \quad \text{(Rule 3)}$$

These rules adjust the likelihood of assigning a field type t_i to a mention candidate $\mathbf{m}$ based on model outputs. The probability of $Me^i(\mathbf{d}, \mathbf{m})$ increases as more models predict t_i, while the probability decreases when models predict alternative labels.

Integration of coarse grained models We introduce an unobserved predicate $Fi^i(\mathbf{d}, \mathbf{v})$ for each field type t_i in T, where $\mathbf{d}$ is a constant representing a document and $\mathbf{m}$ a field candidate in F^i. This predicate indicates the true field mention prediction for each candidate y_{ji}.

Let u^r_{ci} denote the output of a prediction model U^r for a specific label t_i and a field candidate v_c in F^i with $c \in \{1, 2, \ldots, |N|\}, \exists r \in \{1, 2, \ldots, |R|\}, k^r_c = t_i\}$.

For models producing a predicted label for each word, we define $u^r_{ci} = \mathbf{1}\left(k^r_c = t_i\right)$. $\mathbf{1}(\cdot)$ is an indicator function that outputs 1 if the label assigned by the model to value v_c is t_i, and 0 otherwise.

In the case of models outputting a probability distribution over all labels $t_i \in T$, we define u^r_c as the probability assigned by U^q to value v_c and label t_i: $u^r_{ci} = P(t_i \mid v_c, U^q)$.

We introduce an observed predicate $UF^{q,i}(\mathbf{d}, \mathbf{f})$ for each model in U, representing the output u^{qi} for field type t_i, with a constant $\mathbf{d}$ for every document and $\mathbf{f}$ for every field candidate in F^i.

Rule 4 and Rule 5 correlate the outputs of the models U with the unobserved field prediction $Fi^i(\mathbf{d}, \mathbf{f})$ for each field type t_i and model U^r:

$$w^{li}_3 : UF^{ri}(\mathbf{d}, \mathbf{f}) \Rightarrow Fi^i(\mathbf{d}, \mathbf{f}) \quad \text{(Rule 4)}$$

$$w^{li}_4 : !UF^{ri}(\mathbf{d}, \mathbf{f}) \Rightarrow !Fi^i(\mathbf{d}, \mathbf{f}) \quad \text{(Rule 5)}$$

These rules increase the likelihood of assigning a field type t_i to a field candidate as more models predict this label, and conversely, decrease the probability if models predict a different field type.

Linking Fine and Coarse Grained Models As outlined in Section 3, the task involves predicting the existence of a field link a_{jc} for each pair of field mention and field candidate in F_i and M_i. Rather than treating this as an unobserved variable, we define a linking function with values in $[0, 1]$, indicating the likelihood that a field mention candidate belongs to a field candidate. This function can leverage string or embedding similarity, normalized to $[0, 1]$. We adopt the Jaccard distance to compare string representations of field mentions and candidates (refer to Appendix A). To capture the interdependence between field mentions and fields, we introduce Rule 6 and Rule 7. These rules ensure that a field candidate and its mentions share the same field type t_i, enabling bidirectional propagation of predicted field types:

$$w_5 : Me^i(\mathbf{d}, \mathbf{m}) \wedge Lnk(\mathbf{d}, \mathbf{m}, \mathbf{f}) \Rightarrow Fi^i(\mathbf{d}, \mathbf{f}) \quad \text{(Rule 6)}$$

$$w_6 : Fi^i(\mathbf{d}, \mathbf{f}) \wedge Lnk(\mathbf{d}, \mathbf{m}, \mathbf{f}) \Rightarrow Me^i(\mathbf{d}, \mathbf{m}) \quad \text{(Rule 7)}$$

4.4 Cardinality Constraints

In many domains, field types are subject to cardinality constraints. For instance, business documents often contain a unique identification number and multiple entries for fields such as addresses. We extend the information schema by associating each field type t_i with a cardinality cardinality_i and specifying whether the field is mandatory or optional. For mandatory fields, Rule 8 enforces that exactly cardinality_i fields of type t_i are extracted per document. For optional fields, Rule 9 ensures that the number of instances does not exceed cardinality_i.

$$w_7^i : F^i(\mathbf{d}, +\mathbf{f}) = \text{cardinality}_i \quad \text{(Rule 8)}$$

$$w_8^i : F^i(\mathbf{d}, +\mathbf{f}) \leq \text{cardinality}_i \quad \text{(Rule 9)}$$

Here, $F^i(\mathbf{d}, +\mathbf{f})$ denotes the summation over all extracted fields of type t_i in document $\mathbf{d}$.

4.5 LLM as a Judge

Common model errors in information extraction include format inconsistencies, such as partial date entries (e.g., missing years), and factual inaccuracies, such as misidentifying a rural postbox as a headquarters address. To systematically verify and refine field candidates F^i for each field type t_i, we propose utilizing a LLM as a scoring mechanism.

An in-context-tuned LLM assigns a score $s(f) \in [0, 1]$ to each field candidate in $f \in F^i$, representing its likelihood of being a valid instance based on domain specific propertiees such as format adherence and factual correctness. The scoring functions are integrated using Rule 10 and Rule 11:

$$w_9^{li} : LLM^{ri}(\mathbf{d}, \mathbf{f}) \Rightarrow F^i(\mathbf{d}, \mathbf{f}) \quad \text{(Rule 10)}$$

$$w_1^{li}0 : \neg LLM^i(\mathbf{d}, \mathbf{f}) \Rightarrow \neg F^i(\mathbf{d}, \mathbf{f}) \quad \text{(Rule 11)}$$

$LLM^i(\mathbf{d}, \mathbf{f})$ denotes the predicate indicating that the LLM supports field assignment for f based on document d. These rules increase confidence in F^i when $s(f)$ is high and reduce confidence otherwise.

The scoring mechanism is extensible to other feedback functions, such as human annotation scores, regular expressions and database lookups for fact-checking. These feedback signals can be integrated into the meta-model framework to refine the prediction of field candidates F^i improving the overall robustness of the model.

4.6 Learning and Inference

We define a probability distribution over our unobserved random variables—field mentions and fields—conditioned on the observed variables: the prediction outputs from fine-grained models (word and label sequences) and coarse-grained models (key-value pairs), as well as the links between candidates and LLM-judgment scores. The potential functions that govern these distributions are defined by the logical rules, as shown in Formula 3.

The weights ω in the potential functions (e.g., rule weights) are learned by maximizing the likelihood of the observed data under the model, which can be formulated as an optimization problem that maximizes the log-likelihood or another suitable performance metric. During inference, the most probable field mention and field assignment are identified by maximizing the joint probability distribution as specified in Formula 3.

5 Experiments

5.1 Experimental Setup

We evaluate PM3-KIE on the VRDU corpus and compare it against three state-of-the-art models: LayoutLMv3, Lilt, and a fine-tuned GPT-3.5 Turbo (Radford and Narasimhan, 2018; OpenAI, 2024), along with a stacked ensemble (Wolpert, 1992). Additionally, an ablation study assesses the impact of PM3-KIE components on overall performance. A complete list of all software components used for the experiments including their licenses can be found in Appendix E.

Data We use the VRDU dataset, which contains complex ad buy invoices with 9 fields: 'advertiser', 'contract_num', 'gross_amount', 'property', 'tv_address', 'flight_from', 'flight_to', 'product', and 'agency'. The dataset includes different train, development, and test folds, with both in-distribution (ID) and out-of-distribution (OOD) test sets. We evaluate performance using splits of varying training sizes (10, 50, 100, 200), with one ID and one OOD fold and three splits per (train size, distribution)-combination, resulting in 24 models enabling to assess model performance under different training sizes and distribution shifts. We also corrected errors in annotations, especially for date fields. Detail can be found in Appendix F.2 and F.

Evaluation Metrics We apply field-specific matching functions (Wang et al., 2023c) to normalize values and introduce additional functions to address mismatches (see Appendix F.2). As noted by Nourbakhsh et al. (2024), document-level metrics provide a more practical measure of workload

for correcting document extractions than field-level F1 scores. To offer comprehensive insights, we report the metrics in Table 1 Statistical significance is evaluated using paired differences and 95% confidence intervals, calculated with paired standard errors as proposed by Miller (2024). Both are computed for each split and training-size/distribution combination, then averaged across splits to derive confidence intervals.

Metric	Description
F1 per Field	Mean F1 across fields.
F1 per Doc	Mean F1 across documents.
Hit per Doc	Mean hit rate, "hit" = correct field extraction.
Doc-Level Accuracy	perc. of documents with correct extraction

Table 1: Performance Metrics for KIE

Baselines We compare PM3-KIE with three fine-tuned models: LayoutLMv3, Lilt, and GPT-3.5 Turbo. These models not only serve as benchmarks but also as prediction models for PM3-KIE, as described in Section 4. All models are fine-tuned on the training fold and hyperparameters are optimized on the validation fold. Additionally, we compare with a logistic regression stacked ensemble that combines all baseline models, Lilt, LayoutLMv3, and GPT-3.5 Turbo and is trained on the validation fold. Further details on model architectures, prompt templates, training, and hyperparameters are listed in Appendix B.

PM3-KIE Model The final PM3-KIE model integrates all components described in Section **??**. It combines all baseline models, Lilt, LayoutLMv3, and GPT-3.5 Turbo, along with cardinality constraints specifying mandatory fields (contract number, gross amount, advertiser, property) and optional fields (tv address, product, agency, flight start and end date) with at most one field value per document. We incorporate two LLM-based components for fact- and format-checking using GPT-4 mini via the OpenAI API (see G for details about the prompt templates used). Training is conducted on the development fold (details in Appendix C).

5.2 Results and Analysis

Baseline Comparison Table 2 summarizes the F1 scores for PM3-KIE and baseline models, with standard errors in brackets. The "best model" baseline is selected per split as the model achieving the highest F1 score. Results are averaged over three splits for each training size and distribution type. The "paired difference" column shows the performance difference between PM3-KIE and the best baseline, with 95% confidence intervals in brackets. Results demonstrate that PM3-KIE consistently outperforms all baselines across all data chunks with statistical significance (95%), highlighting its robustness in a variety of settings.

Document-Level Metrics To assess practical utility, we analyze performance at the document level, providing insights into error rates per document. Table 3 shows that PM3-KIE achieves significant improvements in document-level accuracy, reducing the practical manual correction workload. This indicates the model's capability to fully automate invoice processing for over 60% of the documents and additional 9% compared to the best performing baseline model.

Impact of Training Size and Document Distribution We investigate whether PM3-KIE performs particularly well under specific conditions, such as limited training data or test folds with unseen document formats. Figure 2 visualizes the performance gains by training size and test distribution type (ID vs. OOD). Gains are averaged over three splits per configuration. The results show that PM3-KIE excels especially in low-resource settings and on OOD test sets. This demonstrates its robustness to distributional shifts and effectiveness in scenarios with limited labeled data.

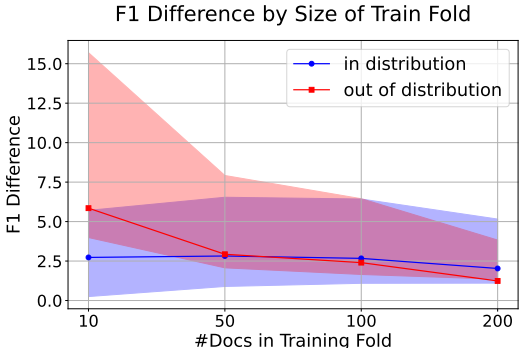

Figure 2: F1 difference (F1 PM3-SKIE and F1 Best Model) with 95% confidence interval by train size for both id and ood test folds.

5.3 Ablation Study

We conduct an ablation study to assess the contributions of individual model components. Models are trained on three splits for each combination of training size (10, 50, 100, 200) and distribution type (ID

Table 2: Performance metrics for various training sizes (Train) and OOD test sets. The reported F1 scores are averaged across all fields and are accompanied by their corresponding standard errors in parentheses. The paired difference between PM3-KIE and the best-performing model is presented, along with the 95% confidence interval enclosed in parentheses. A detailed result table with ID and OOD test set results is displayed in Appendix D

Dist	#Train	#Test	F1 LILT (std. error)	F1 GPT (std. error)	F1 LMv3 (std. error)	F1 STE (std. error)	F1 Best (std. error)	F1 PM3-KIE (std. error)	Paired Diff (conf. interval)
ood	10	294	64.89	78.00	58.11	72.13	78.00	**82.29**	**+5.85**
			(1.50)	(1.48)	(1.60)	(1.15)	(1.48)	(1.15)	(+1.86, +9.85)
ood	50	236	89.86	85.65	83.46	84.53	89.86	**92.44**	**+2.93**
			(0.97)	(1.42)	(1.21)	(1.07)	(0.97)	(0.83)	(+0.86, +4.99)
ood	100	211	91.10	86.76	83.66	84.01	91.10	**92.99**	**+2.40**
			(1.07)	(1.21)	(1.28)	(1.04)	(1.07)	(0.88)	(+0.75, +4.05)
ood	200	156	93.11	90.19	85.30	88.40	93.11	**94.38**	**+1.24**
			(0.99)	(1.46)	(1.41)	(1.06)	(0.99)	(0.96)	(+0.11, +2.59)

Table 3: Mean F1-scores, Hit-rate and Accuracy per Field and per Document averaged over all 24 models

Model	F1 per Field	Hit per Doc	F1 per Doc	Doc Lv. Acc.
LILT	88.24	87.40	86.85	48.84
LMv3	81.45	80.10	79.48	24.69
GPT3.5	87.60	88.81	86.28	53.35
Stacked	84.64	79.15	83.13	19.03
Best Model	90.22	89.86	88.83	53.55
PM3-SKIE	**92.47**	**92.28**	**91.67**	**62.38**

Table 4: Ablation study results. Difference in F1 for the basic PM3-KIE model compared to the adapted model version.

Ablation	F1 Field	Hit Doc	F1 Doc	Acc
PM3-KIE	**92.47**	**92.28**	**91.67**	**62.38**
+ Token Task	-0.59	+0.23	-0.41	+0.18
- Constraints	-1.88	-1.76	-1.76	-7.78
- fact & format judges	-0.28	-0.34	-0.26	-2.03
- fact judge	-0.17	-0.12	-0.15	-0.62
- format judge	-0.21	-0.31	-0.18	-1.75

and OOD). Table 4 summarizes the findings.

The basic model is a model learned only on the final field extraction data with two LLM judges for fact and format checking and with information schema constraints.

Granular Labeling. To assess the impact of providing granular labels at the field mention level, we compare a multitask learning approach ("+Token Task") with the standard single-task setup, where ground truth is only available for field extractions. The results indicate that multitask learning yields only marginal improvements in hit-rate per document and document-level accuracy, while slightly reducing performance in other metrics.

Cardinality Constraints. We assess the role of cardinality constraints by training models without these constraints ("-constraints"). Performance metrics decrease across all metrics, confirming that these constraints enhance model performance.

Effectiveness of LLM-as-a-Judge. Two LLMs for fact-checking and format-checking are integrated into PM3-KIE. Removing these components ("- fact & format judges") slightly reduces F1 per Field, F1 per Document, and Hit Rate by approximately 0.3%, but document-level accuracy drops significantly by 2%, highlighting the judges' importance for precise field extractions. We compare against removing only one of the two LLMs ("-fact judge" and "-format judge"). These perform slightly better then removing both, but still worse than the base model. This underscores the advantage of each of the task-specific LLM judges.

6 Conclusion and Discussion

In this work, we have proposed PM3-KIE, a probabilistic multi-task meta-model that addresses the challenges in KIE. By integrating both fine-grained token classification models and coarse-grained entity extraction models, our approach provides a flexible and robust solution for handling diverse KIE tasks. Through the incorporation of schema consistency and LLM-based validation, PM3-KIE ensures logical adherence to extraction schemas and semantic plausibility, significantly reducing errors and improving reliability. Our experiments on the VRDU dataset demonstrate that PM3-KIE outperforms existing state-of-the-art methods across a range of scenarios, including in-distribution, out-of-distribution, and low-resource conditions. These results highlight the potential of our approach in document processing tasks.

7 Limitations

Prompt Design Sensitivity: The performance of the LLM-as-a-judge component is sensitive to both the capability of the LLM (e.g., parameter

size and training data) as well as the quality of the prompts used. For example, the capability for fact checking requires up to data training data that contain the fact. Inadequate model size or poorly designed prompts may lead to unreliable validation scores While the meta-models training ensures that weights are adjusted in a way to minimize the effect of uninformative LLM outputs, they will not benefit overall performance in such cases. In future we plan to show that in form of additional robustness checks with noisy models.

Dependency on Base Models: The extraction quality of the meta-model relies on the performance of the base fine-grained and coarse-grained models. If all base models fail to detect a true extraction as potential candidates, they cannot be identified by the overall system. This limitation is common for most ensembling and meta-model approaches.

Computational Costs: While PM3-KIE itself is lightweight with relatively few parameters, the meta-model's complexity arises from integrating fine-grained and coarse-grained models along with LLMs for validation. This integration increases both computational overhead and deployment complexity, as it requires managing multiple models in conjunction with the meta-model. This is a common challenge in ensemble and meta-model approaches.

Additional Processing: PM3-KIE requires data to be formatted as specified in Section 4, including the creation of constants for predicates and truth values for observed predicates. This necessitates additional computational effort for postprocessing both model outputs and input data to meet the required format.

Closed-World Assumption: PM3-KIE assumes all fields to be extracted are known in advance, limiting its applicability to scenarios involving the detection of new or unknown field types.

Dependency on Parsed Strings: This work assumes input documents are in a machine-readable format, typically processed through OCR. OCR error correction and parsing accuracy are beyond the scope of this study, with the approach presuming that such errors have been corrected prior to downstream processing.

8 Ethical Considerations

Automation and Job Displacement: Automating key information extraction from documents, especially in business-critical domains like finance and legal, could reduce the demand for manual data entry and administrative roles. While this improves efficiency and reduces operational costs, it risks unemployment for workers currently performing these tasks.

Risk of Overreliance on Automated Systems: Deploying PM3-KIE in critical sectors, such as healthcare, legal documentation, or property records, may lead to errors being accepted without human verification. Incorrect or incomplete extractions could have significant consequences, including legal disputes, financial losses, or medical errors. PM3-KIE should always operate in a semi-automated manner with manual review.

Bias and Fairness Concerns: Like many AI systems, PM3-KIE's performance depends on the quality and diversity of training data used to train the base models integrated. Biases in the training data could lead to unequal performance across document types, languages, or regions, potentially disadvantaging users from underrepresented groups. Care must be taken to curate balanced datasets and evaluate the model across diverse scenarios.

References

Srikar Appalaraju, Bhavan Jasani, Bhargava Urala Kota, Yusheng Xie, and R. Manmatha. 2021. Docformer: End-to-end transformer for document understanding. In *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, page 973–983. IEEE.

Stephen H. Bach, Matthias Broecheler, Bert Huang, and Lise Getoor. 2017. Hinge-loss markov random fields and probabilistic soft logic. *J. Mach. Learn. Res.*, 18(1):3846–3912.

Haoli Bai, Zhiguang Liu, Xiaojun Meng, Li Wentao, Shuang Liu, Yifeng Luo, Nian Xie, Rongfu Zheng, Liangwei Wang, Lu Hou, Jiansheng Wei, Xin Jiang, and Qun Liu. 2023. Wukong-reader: Multi-modal pre-training for fine-grained visual document understanding. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 13386–13401, Toronto, Canada. Association for Computational Linguistics.

Dário Augusto Borges Oliveira and Matheus Palhares Viana. 2017. Fast cnn-based document layout analysis. In *2017 IEEE International Conference on Computer Vision Workshops (ICCVW)*, pages 1173–1180.

Panfeng Cao, Ye Wang, Qiang Zhang, and Zaiqiao Meng. 2023. GenKIE: Robust generative multimodal document key information extraction. In *The 2023 Conference on Empirical Methods in Natural Language Processing*.

Franz Louis Cesista, Rui Aguiar, Jason Kim, and Paolo Acilo. 2024. Retrieval augmented structured generation: Business document information extraction as tool use. *2024 IEEE 7th International Conference on Multimedia Information Processing and Retrieval (MIPR)*, pages 227–230.

Jiayi Chen, Hanjun Dai, Bo Dai, Aidong Zhang, and Wei Wei. 2023. On task-personalized multimodal few-shot learning for visually-rich document entity retrieval. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 9006–9025, Singapore. Association for Computational Linguistics.

Laura Chiticariu, Yunyao Li, and Frederick R. Reiss. 2013. Rule-based information extraction is dead! long live rule-based information extraction systems! In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 827–832, Seattle, Washington, USA. Association for Computational Linguistics.

B. Cohen and M. York. 2020. Ardent partners' accounts payable metrics that matter in 2020. technical report. *Ardent Partners (2020)*.

Timo I. Denk and Christian Reisswig. 2019. Bertgrid: Contextualized embedding for 2d document representation and understanding. *NeurIPS*, abs/1909.04948.

Yihao Ding, Lorenzo Vaiani, Soyeon Caren Han, Jean Lee, Paolo Garza, Josiah Poon, and Luca Cagliero. 2024. 3mvrd: Multimodal multi-task multi-teacher visually-rich form document understanding. In *Annual Meeting of the Association for Computational Linguistics*.

Łukasz Garncarek, Rafał Powalski, Tomasz Stanisławek, Bartosz Topolski, Piotr Halama, Michał Turski, and Filip Graliński. 2021. *LAMBERT: Layout-Aware Language Modeling for Information Extraction*, page 532–547. Springer International Publishing.

Girsch-Bock, Mary. 2024. Invoice processing cost: What is it, how to calculate it, and how to reduce it. Reviewed on 26 November 2024.

Jiabang He, Lei Wang, Yi Hu, Ning Liu, Hui Liu, Xing Xu, and Heng Tao Shen. 2023. Icl-d3ie: In-context learning with diverse demonstrations updating for document information extraction. In *2023 IEEE/CVF International Conference on Computer Vision (ICCV)*, page 19428–19437. IEEE.

Yupan Huang, Tengchao Lv, Lei Cui, Yutong Lu, and Furu Wei. 2022. Layoutlmv3: Pre-training for document ai with unified text and image masking. In *Proceedings of the 30th ACM International Conference on Multimedia*, MM '22. ACM.

Zheng Huang, Kai Chen, Jianhua He, Xiang Bai, Dimosthenis Karatzas, Shijian Lu, and C. V. Jawahar. 2019. Icdar2019 competition on scanned receipt ocr and information extraction. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*. IEEE.

Wonseok Hwang, Jinyeong Yim, Seunghyun Park, Sohee Yang, and Minjoon Seo. 2020. Spatial dependency parsing for semi-structured document information extraction. In *Findings*.

Anoop R Katti, Christian Reisswig, Cordula Guder, Sebastian Brarda, Steffen Bickel, Johannes Höhne, and Jean Baptiste Faddoul. 2018. Chargrid: Towards understanding 2D documents. *EMNLP*, pages 4459–4469.

Geewook Kim, Teakgyu Hong, Moonbin Yim, JeongYeon Nam, Jinyoung Park, Jinyeong Yim, Wonseok Hwang, Sangdoo Yun, Dongyoon Han, and Seunghyun Park. 2022. *OCR-Free Document Understanding Transformer*, page 498–517. Springer Nature Switzerland.

Chen-Yu Lee, Chun-Liang Li, Timothy Dozat, Vincent Perot, Guolong Su, Nan Hua, Joshua Ainslie, Renshen Wang, Yasuhisa Fujii, and Tomas Pfister. 2022. FormNet: Structural encoding beyond sequential modeling in form document information extraction. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 3735–3754, Dublin, Ireland. Association for Computational Linguistics.

Chen-Yu Lee, Chun-Liang Li, Chu Wang, Renshen Wang, Yasuhisa Fujii, Siyang Qin, Ashok Popat, and Tomas Pfister. 2021. Rope: Reading order equivariant positional encoding for graph-based document information extraction. In *Annual Meeting of the Association for Computational Linguistics*.

Yulin Li, Yuxi Qian, Yuechen Yu, Xiameng Qin, Chengquan Zhang, Yan Liu, Kun Yao, Junyu Han, Jingtuo Liu, and Errui Ding. 2021. Structext: Structured text understanding with multi-modal transformers. In *Proceedings of the 29th ACM International Conference on Multimedia*, MM '21, page 1912–1920. ACM.

Jerry Liu. 2022. LlamaIndex.

Xiaojing Liu, Feiyu Gao, Qiong Zhang, and Huasha Zhao. 2019. Graph convolution for multimodal information extraction from visually rich documents. *NAACL-HLT*, pages 32–39.

Chuwei Luo, Yufan Shen, Zhaoqing Zhu, Qi Zheng, Zhi Yu, and Cong Yao. 2024. Layoutllm: Layout instruction tuning with large language models for document understanding. In *2024 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 15630–15640. IEEE.

Bodhisattwa Prasad Majumder, Navneet Potti, Sandeep Tata, James Bradley Wendt, Qi Zhao, and Marc Najork. 2020. Representation learning for information extraction from form-like documents. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6495–6504, Online. Association for Computational Linguistics.

Evan Miller. 2024. Adding error bars to evals: A statistical approach to language model evaluations.

Armineh Nourbakhsh, Sameena Shah, and Carolyn Rose. 2024. Towards a new research agenda for multimodal enterprise document understanding: What are we missing? In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 14610–14622, Bangkok, Thailand. Association for Computational Linguistics.

OpenAI. 2023. Gpt-4 turbo. Accessed: December 2024.

OpenAI. 2024. *GPT-3.5 Turbo*. Accessed: 2024-12-05.

Rasmus Berg Palm, Ole Winther, and Florian Laws. 2017. Cloudscan - a configuration-free invoice analysis system using recurrent neural networks. In *2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR)*, page 406–413. IEEE.

Qiming Peng, Yinxu Pan, Wenjin Wang, Bin Luo, Zhenyu Zhang, Zhengjie Huang, Yuhui Cao, Weichong Yin, Yongfeng Chen, Yin Zhang, Shikun Feng, Yu Sun, Hao Tian, Hua Wu, and Haifeng Wang. 2022. ERNIE-layout: Layout knowledge enhanced pre-training for visually-rich document understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2022*, pages 3744–3756, Abu Dhabi, United Arab Emirates. Association for Computational Linguistics.

Vincent Perot, Kai Kang, Florian Luisier, Guolong Su, Xiaoyu Sun, Ramya Sree Boppana, Zilong Wang, Zifeng Wang, Jiaqi Mu, Hao Zhang, Chen-Yu Lee, and Nan Hua. 2024. LMDX: Language model-based document information extraction and localization. In *Findings of the Association for Computational Linguistics: ACL 2024*, pages 15140–15168, Bangkok, Thailand. Association for Computational Linguistics.

Rafał Powalski, Łukasz Borchmann, Dawid Jurkiewicz, Tomasz Dwojak, Michał Pietruszka, and Gabriela Pałka. 2021. *Going Full-TILT Boogie on Document Understanding with Text-Image-Layout Transformer*, page 732–747. Springer International Publishing.

Yujie Qian, Enrico Santus, Zhijing Jin, Jiang Guo, and Regina Barzilay. 2019. Graphie: A graph-based framework for information extraction. In *Proceedings of the 2019 Conference of the North (NAACL-HLT)*, page 751–761. Association for Computational Linguistics.

Alec Radford and Karthik Narasimhan. 2018. Improving language understanding by generative pre-training.

Clément Sage, Alexandre Aussem, Haytham Elghazel, Véronique Eglin, and Jérémy Espinas. 2019. Recurrent neural network approach for table field extraction in business documents. In *2019 International Conference on Document Analysis and Recognition (ICDAR)*, pages 1308–1313.

Daniel Schuster, Klemens Muthmann, Daniel Esser, Alexander Schill, Michael Berger, Christoph Weidling, Kamil Aliyev, and Andreas Hofmeier. 2013. Intellix – end-user trained information extraction for document archiving. *2013 12th International Conference on Document Analysis and Recognition*, pages 101–105.

Dengliang Shi, Siliang Liu, Jintao Du, and Huijia Zhu. 2023. Layoutgcn: A lightweight architecture for visually rich document understanding. In *IEEE International Conference on Document Analysis and Recognition*.

Zineng Tang, Ziyi Yang, Guoxin Wang, Yuwei Fang, Yang Liu, Chenguang Zhu, Michael Zeng, Cha Zhang, and Mohit Bansal. 2023. Unifying vision, text, and layout for universal document processing. In *2023 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, page 19254–19264. IEEE.

Yi Tu, Ya Guo, Huan Chen, and Jinyang Tang. 2023. Layoutmask: Enhance text-layout interaction in multi-modal pre-training for document understanding. In *Annual Meeting of the Association for Computational Linguistics*.

Dongsheng Wang, Natraj Raman, Mathieu Sibue, Zhiqiang Ma, Petr Babkin, Simerjot Kaur, Yulong Pei, Armineh Nourbakhsh, and Xiaomo Liu. 2023a. Docllm: A layout-aware generative language model for multimodal document understanding. *Preprint*, arXiv:2401.00908.

Jiapeng Wang, Lianwen Jin, and Kai Ding. 2022. Lilt: A simple yet effective language-independent layout transformer for structured document understanding. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, page 7747–7757. Association for Computational Linguistics.

Zifeng Wang, Zizhao Zhang, Jacob Devlin, Chen-Yu Lee, Guolong Su, Hao Zhang, Jennifer Dy, Vincent Perot, and Tomas Pfister. 2023b. QueryForm: A simple zero-shot form entity query framework. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 4146–4159, Toronto, Canada. Association for Computational Linguistics.

Zilong Wang, Yichao Zhou, Wei Wei, Chen-Yu Lee, and Sandeep Tata. 2023c. Vrdu: A benchmark for visually-rich document understanding. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, KDD '23. ACM.

Mengxi Wei, Yifan He, and Qiong Zhang. 2020. Robust layout-aware ie for visually rich documents with pre-trained language models. *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2019. Huggingface's transformers: State-of-the-art natural language processing. *Preprint*, arXiv:1910.03771.

David Wolpert. 1992. Stacked generalization. *Neural Networks*, 5:241–259.

Xiao Yang, Ersin Yumer, Paul Asente, Mike Kraley, Daniel Kifer, and C. Lee Giles. 2017. Learning to extract semantic structure from documents using multimodal fully convolutional neural networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, page 4342–4351. IEEE.

Minghong Yao, Zhiguang Liu, Liansheng Zhuang, Liangwei Wang, and Houqiang Li. 2024. A robust framework for one-shot key information extraction via deep partial graph matching. *IEEE Transactions on Image Processing*, 33:1070–1079.

Wenwen Yu, Ning Lu, Xianbiao Qi, Ping Gong, and Rong Xiao. 2020. Pick: Processing key information extraction from documents using improved graph learning-convolutional networks. *2020 25th International Conference on Pattern Recognition (ICPR)*, pages 4363–4370.

Yuechen Yu, Yulin Li, Chengquan Zhang, Xiaoqiang Zhang, Zengyuan Guo, Xiameng Qin, Kun Yao, Junyu Han, Errui Ding, and Jingdong Wang. 2023. Structextv2: Masked visual-textual prediction for document image pre-training. In *ICLR*. OpenReview.net.

Peng Zhang, Yunlu Xu, Zhanzhan Cheng, Shiliang Pu, Jing Lu, Liang Qiao, Yi Niu, and Fei Wu. 2020. Trie: End-to-end text reading and information extraction for document understanding. *Proceedings of the 28th ACM International Conference on Multimedia*.

Xiaohui Zhao, Zhuo Wu, and Xiaoguang Wang. 2019. CUTIE: learning to understand documents with convolutional universal text information extractor. *ICDAR*, abs/1903.12363.

A Similarity Score Calculation

To match field mentions to fields, a similarity score, displayed in Algorithm 1, is calculated based on their string representations.

Algorithm 1 Linking Function

Require: token_str, field_str
Ensure: link

function `calculate_link(token_str, field_str)`
if token_str.isnumeric() and token_str = field_str **then**
  return 1.0
else if ¬token_str.isnumeric() **then**
  sim ← jaccard_similarity(token_str, field_str)
  if sim > 0.7 **then**
    return sim
  else
    return 0.0
  end if
end if
end function

function `jaccard_similarity(text1, text2)`
if len(text1) = 0 or len(text2) = 0 **then**
  return 0
end if
if len(text1) < 4 and len(text2) < 4 **then**
  if text1 = text2 **then**
    return 1.0
  else
    return 0.0
  end if
end if
$n1 \leftarrow$ ngrams(text1, 3)
$n2 \leftarrow$ ngrams(text2, 3)
jsim $\leftarrow 1 -$ jaccard_distance(set($n1$), set($n2$))
jsim $\leftarrow \max(0, \text{jsim} - 0.1 \times |\text{len(text1)} - \text{len(text2)}|)$
return jsim
end function

B Baseline Training Details

Stacked Ensemble: The stacked ensemble model utilizes the Logistic Regression implementation from `sklearn`[1] and during training, hyperparameters are tuned with a grid search over the following hyperparameters: {'C': [0.01, 0.1, 1, 10, 100, 1000], 'penalty': ['l1', 'l2'], 'solver':['liblinear', 'saga']}. For each field candidate, we construct an input vector comprising features that include prediction scores for each field type and model. Field mentions detected by LayoutLMv3 and LiLT are mapped to field candidates during preprocessing using the similarity algorithm detailed in Appendix 1. For the GPT-3 model, prediction scores are binary (0,1), whereas LiLT and LayoutLMv3 provide predicted probabilities for each field candidate. Additionally, LLM judge scores for factual and format correctness per field type are incorporated as features.

Token Sequence Tagging Task LiLT and LayoutLMv3 are fine-tuned on a downstream sequence

[1] https://scikit-learn.org/1.5/modules/generated/sklearn.linear_model.LogisticRegression.html

tagging task with a classification head for token classification, utilizing the BILOU schema. However, certain fields span multiple token sequences within the document, making them incompatible with the BILOU schema. To address this, we introduce an additional label, `LABELNAME_ADD`, for subsequent field sequences following the first. In the DeepForm VRDU dataset, this applies to "tv_address" and "product" fields. During training, the models predict these `_ADD` labels, and in postprocessing, scattered fields are reconstructed by linking each `_ADD`-labeled sequence to its corresponding sequence labeled without `_ADD`.

LiLT: The LiLT model was trained and applied using the `LayoutLMv3TokenizerFast`, `AutoModelForTokenClassification`, and `Trainer` classes from the Transformers library (Wolf et al., 2019)`https://huggingface.co/docs/transformers/index`.

We utilized the pretrained tokenizer and model `SCUT-DLVCLab/lilt-roberta-en-base`, provided by the authors (Wang et al., 2022) on Hugging Face[2]. The tokenizer was configured with the following settings: `truncation=True`, `stride=128`, `padding="max_length"`, `max_length=512`, `return_overflowing_tokens=True`, and `return_offsets_mapping=True`.

The model was fine-tuned using the `AutoModelForTokenClassification` class and the `Trainer`, with the following hyperparameters:

- Learning rate: 5×10^{-6}
- Batch size: 8
- Gradient accumulation steps: 4
- Maximum steps: 9000
- Metric for model selection: `overall_f1`
- Warmup ratio: 0.1

The experiments were conducted on a system equipped with dual Intel Xeon Gold 6226R CPUs (64 cores, 128 threads), 754 GB of RAM, and two NVIDIA Tesla V100 32GB GPUs. Each experiment utilized a single GPU and required approximately 20 hours to complete.

[2] `https://huggingface.co/SCUT-DLVCLab/lilt-roberta-en-base`

LayoutLMv3: The LayoutLMv3 model was trained and applied using the `AutoProcessor`, `LayoutLMv3ForTokenClassification`, and `Trainer` classes from the Transformers library.

We utilized the pretrained tokenizer and model `microsoft/layoutlmv3-base`, provided by the authors (Huang et al., 2022) on Hugging Face[3]. The processor was loaded and applied using the `AutoProcessor` class with the setting `apply_ocr=False`.

The model was fine-tuned using the `LayoutLMv3ForTokenClassification` class and the `Trainer` with the following hyperparameters:

- Metric for model selection: `overall_f1`
- Warmup ratio: 0.1
- Learning rate: 5×10^{-6}
- Batch size: 8
- Gradient accumulation steps: 4
- Maximum steps: 9000

The experiments were conducted on a system equipped with dual Intel Xeon Gold 6226R CPUs (64 cores, 128 threads), 754 GB of RAM, and two NVIDIA Tesla V100 32GB GPUs. Each experiment utilized a single GPU and required approximately 20 hours to complete.

GPT-3.5: The GPT-3.5 Turbo model (OpenAI, 2024) is a decoder-based large language model (LLM) fine-tuned for the entity extraction task, following the tool-use approach introduced by Cesista et al. (2024). Unlike Cesista et al. (2024), we do not employ structured prompting to transform PDF content but instead use raw PDF text to minimize additional processing costs.

For supervised fine-tuning, we utilize the OpenAI plattform[4] to fine-tune the `gpt-3.5-turbo-0125` model. The `gpt-3.5-turbo-0125` model was chosen as it is more cost-effective than newer models while maintaining a knowledge cutoff in 2022, ensuring that the base model has not been exposed to the VRDU dataset.

[3] `https://huggingface.co/microsoft/layoutlmv3-base`

[4] `https://platform.openai.com/finetune`

The fine-tuning process follows the guidelines provided in the OpenAI Cookbook[5], and the resulting model generates valid JSON outputs. These outputs are parsed to extract entities for downstream tasks. Training costs amounted to approximately €210 ($221), with additional application costs for the test sets estimated at €75 ($79).

We implement retrieval-augmented generation using OpenAI's ChatCompletion API with Function Calling[6] in conjunction with the Python package `llama-index`(Liu, 2022). Fine-tuning was conducted following the instructions available in the OpenAI Cookbook[7]. The function definition, detailed in Listing 1, defines the data schema for invoice fields and is passed to the ChatCompletion API as a tool[8].

The model produces valid JSON outputs, from which entities are extracted for further processing. The cost of training the model was approximately €210 ($221), and applying the model to the test sets incurred an additional cost of approximately €75 ($79).

The system prompt utilized for processing political advertisement invoice documents is shown in Listing 2. This prompt, combined with the function dictionary (Listing 1), is supplied to the OpenAI ChatCompletion API to facilitate fine-tuning and inference.

To align the assistant's responses with ground truth extractions from the VRDU dataset, these extractions are provided as examples during fine-tuning. The format of these examples is presented in Listing 3.

C MP3-KIE Implementation, Training and Inference Details

Model generation, training, and inference were based on the Statistical Relational Learning framework introduced by Bach et al. (2017). For our experiments, we utilized the `pslpython` package (version 2.4.0[9]), which wraps the Java-based PSL implementation[10].

Meta-model weight learning was performed on the development folds using the structured perceptron algorithm[11]. Inference was conducted using ADMM[12] as described by Bach et al. (2017).

Default hyperparameters were employed, with modifications to the regularization settings as follows:

- `gradientdescent.negativelogregularization = 0`
- `gradientdescent.negativeentropyregularization = 0.0001`

All experiments were executed on a CPU system equipped with 48 cores (Intel® Xeon® Gold 5118, 2.3 GHz base clock, 3.2 GHz max clock) and 376 GiB of RAM. Training the meta-model weights required approximately 20 minutes per model on average.

D In Distribution Results on VRDU

Table 5 summarizes the ID and OOD results of all baselines and PM3-KIE on the VRDU dataset.

E Additional Software and Licenses

Table 6 lists all first-level import python packages used to perform the experiments described in this work.

F Dataset

F.1 VRDU Dataset and Evaluation

We utilized the VRDU dataset and evaluation framework implementation provided by Wang et al. (2023c).[13]

F.2 Annotation and Evaluation Corrections

To reduce spurious matching errors during evaluation, we employed the field-specific matching functions available for the dataset[14] to normalize values, such as standardizing date formats. In addition to the existing functions, we introduced three additional matching strategies to address inconsistencies in the dataset:

- **GeneralCaseInsensitiveStringMatch**: Strings are considered equivalent if their lowercase representations match.

[5] https://cookbook.openai.com/examples/chat_finetuning_data_prep
[6] https://platform.openai.com/docs/guides/function-calling
[7] https://cookbook.openai.com/examples/chat_finetuning_data_prep
[8] https://platform.openai.com/docs/api-reference/chat
[9] https://pypi.org/project/pslpython/
[10] https://github.com/linqs/psl
[11] org.linqs.psl.application.learning.weight.gradient .optimalvalue.StructuredPerceptron
[12] org.linqs.psl.application.inference.mpe.ADMMInference
[13] https://github.com/google-research-datasets/vrdu
[14] https://github.com/google-research/google-research/tree/master/vrdu

Listing 1: Function Description for OpenAI Tool Use

```
function_dict = {
 'name': 'AdInvoice', 'description': 'Data model for invoice fields.', 'parameters':
  {'title': 'AdInvoice', 'description': 'Data model for invoice fields.', 'type': 'object',
   'properties': {
     'contract_num': {
       'title': 'Contract Number',
       'description': 'The invoice contract number or order number.',  'type': 'string'},
     'tv_address': {
       'title': 'Tv Address', 'description': 'Physical address of the tv channel.',
       'default': '', 'type': 'string'},
     'property': {
       'title': 'Property', 'description': 'Property, usually equivalent to tv channel name.',
       'default': '', 'type': 'string'},
     'agency': {
       'title': 'Agency', 'description': 'The advertisement agency.', 'default': '', 'type': 'string'},
     'advertiser': {'title': 'Advertiser', 'description': 'The advertiser.', 'type': 'string'},
     'flight_from': {
       'title': 'Flight From',
       'description': 'The order flight start date.',
       'default': '',
       'type': 'string'},
     'flight_to': {
       'title': 'Flight To',
       'description': 'The order flight end date.',
       'default': '',
       'type': 'string'},
     'product': {
       'title': 'Product',
       'description': 'The product that is advertised.',
       'default': '',
       'type': 'string'},
     'gross_amount': {
       'title': 'Gross Amount',
       'description': 'The total amount to be paid.',
       'type': 'string'},
     'line_items': {
       'title': 'Line Items',
       'description': 'List of line items.',
       'type': 'array',
       'items': {
         '$ref': '#/definitions/LineItem' }}},
   'required': ['contract_num', 'advertiser', 'line_items'],
   'definitions': {
     'LineItem': {
       'title': 'LineItem',
       'description': 'Data model for line item fields.',
       'type': 'object',
       'properties': {
         'channel': {
           'title': 'Channel',
           'description': 'Name of the tv channel broadcasting the advertisement.',
           'default': '',
           'type': 'string'},
         'program_start_date': {
           'title': 'Program Start Date',
           'description': 'Program start date (only date without timestamp).',
           'default': '',
           'type': 'string'},
         'program_end_date': {
           'title': 'Program End Date',
           'description': 'Program end date (only date without timestamp).',
           'default': '',
           'type': 'string'},
         'program_desc': {
           'title': 'Program Desc',
           'description': 'Description of the TV program.',
           'default': '',
           'type': 'string'},
         'sub_amount': {
           'title': 'Sub Amount',
           'description': 'Sub amount for one program ad.',
           'default': '',
           'type': 'string'}}}}}}
```

Listing 2: Formatted System Prompt

```
You are receiving content from a political advertisement invoice document.
This invoice is signed between a TV station and a campaign group.
The document uses tables, multi-columns, and key-value pairs to record the information.
Your task is to digitize these documents by extracting their information in a structured format.
Extract the unique header information, such as TV channel addresses and total costs,
along with the list of line items detailing specific ads, TV programs in which they will be broadcasted,
and sub-amounts.
Extract line item fields only from the tabular line item list in the invoice document
and not from the invoice header: if line item fields are not present in the line item list, don't extract them.
```

Listing 3: Ground Truth Assistant Answer provided for Fine-Tuning


```
{
  "role": "assistant",
  "function_call": {
    "name": "AdInvoice",
    "arguments":
     "{"contract_num":"668864 ",
       "tv_address":"PO Box 809229\\nChicago, IL 60680-9229\\n",
       "property":"WAXN-TV\\nWSOC Television, Inc.\\n",
       "agency":"",
       "advertiser":"POL/Donald Trump/R/PRES/US-A\\n",
       "flight_from":"03/05/20 ",
       "flight_to":"03/10/20\\n",
       "product":"TRUMP FOR PRESIDENT\\n",
       "gross_amount":"$3,920.00\\n",
       "line_items":[
        {"channel":"WAXN ","program_start_date":"03/09/20 ","program_end_date":"03/09/20 ",
        "program_desc":"M-F 7a-8a\\n","sub_amount":"$250.00\\n"},
        {"channel":"WAXN ","program_start_date":"03/06/20 ","program_end_date":"03/06/20 ",
        "program_desc":"M-F 7a-8a\\n","sub_amount":"$250.00\\n"},
        {"channel":"WAXN ","program_start_date":"03/09/20 ","program_end_date":"03/09/20 ",
        "program_desc":"M-F 8a-9a\\n","sub_amount":"$260.00\\n"},
        {"channel":"WAXN ","program_start_date":"03/06/20 ","program_end_date":"03/06/20 ",
        "program_desc":"M-F 8a-9a\\n","sub_amount":"$260.00\\n"},
        {"channel":"WAXN ","program_start_date":"03/06/20 ","program_end_date":"03/06/20 ",
        "program_desc":"M-F 8p-830p\\n","sub_amount":"$0.00\\n"},
        ....,}
    ]}"
  }
}
```


Table 5: Performance metrics for various training set sizes (Train) and distribution folds (Dist). The reported F1 scores are F1-scores averaged across all fields and are accompanied by their corresponding standard errors in parentheses. Additionally, the paired difference between our model and the best-performing model is presented, along with the 95% confidence interval enclosed in parentheses.

Dist	#Train	#Test	F1 LILT (std. error)	F1 GPT (std. error)	F1 LMv3 (std. error)	F1 STE (std. error)	F1 Best (std. error)	F1 PM3-KIE (std. error)	Paired Diff (conf. interval)
iid	10	6	87.36 (4.49)	87.92 (5.14)	80.97 (4.32)	86.84 (4.80)	90.09 (2.65)	**92.79** (2.84)	**+2.73** (+2.48, +2.98)
iid	50	64	91.55 (2.36)	87.92 (2.25)	82.30 (3.12)	85.36 (2.08)	91.55 (2.36)	**93.39** (1.86)	**+2.82** (+1.93, +3.72)
iid	100	89	93.85 (1.66)	92.41 (1.59)	87.72 (2.11)	88.63 (1.48)	93.85 (1.66)	**95.70** (1.17)	**+2.67** (+1.58, +3.76)
iid	200	144	94.21 (1.20)	91.99 (1.64)	90.06 (1.19)	87.21 (1.16)	94.21 (1.20)	**95.81** (0.93)	**+2.03** (+0.93, +3.13)
ood	10	294	64.89 (1.50)	78.00 (1.48)	58.11 (1.60)	72.13 (1.15)	78.00 (1.48)	**82.29** (1.15)	**+5.85** (+1.86, +9.85)
ood	50	236	89.86 (0.97)	85.65 (1.42)	83.46 (1.21)	84.53 (1.07)	89.86 (0.97)	**92.44** (0.83)	**+2.93** (+0.86, +4.99)
ood	100	211	91.10 (1.07)	86.76 (1.21)	83.66 (1.28)	84.01 (1.04)	91.10 (1.07)	**92.99** (0.88)	**+2.40** (+0.75, +4.05)
ood	200	156	93.11 (0.99)	90.19 (1.46)	85.30 (1.41)	88.40 (1.06)	93.11 (0.99)	**94.38** (0.96)	**+1.24** (+0.11, +2.59)

- **IgnorePropertySuffixStringMatch**: Accounts for inconsistent annotations of properties (e.g., with or without the suffix *"remit to"*). Strings are matched after removing the phrase *"remit to"* and redundant whitespace.
- **IgnoreLeadingTrailingNumbersStringMatch**: Handles inconsistent annotations for products and agencies where numeric prefixes or suffixes (including those in brackets) may or may not be present. Strings are matched

Table 6: Software Packages and Their Licenses

Name	Version	License
python	3.10.14	PSF License
editdistance	3.10.14	MIT License
datasets	2.19.0	Apache Software License
pandas	2.2.3	BSD License
scikit-learn	1.5.1	BSD License
tqdm	4.66.4	MIT License; MPL 2.0
pydantic	2.8.2	MIT License
pydantic_core	2.20.1	MIT License
transformers	4.40.1	Apache Software License
numpy	1.26.4	BSD License
torch	2.3.0	BSD License
pillow	10.3.0	HPND
llama-index	0.10.52	MIT License
tiktoken	0.7.0	MIT License
nltk	3.8.1	Apache Software License
pdf2image	1.17.0	MIT License
dataclasses-json	0.6.7	MIT License
mlflow	2.12.2	Apache Software License
plotly	5.22.0	MIT License
matplotlib	3.9.0	Python SF License
scipy	1.13.0	BSD License

after removing leading or trailing numeric sequences and redundant whitespace.

Several annotations in the VRDU dataset were incorrect, incomplete, or missing, particularly for dates and address elements. For instance, dates were often missing complete year information, and address components were inconsistently annotated. Table 7 lists the corrections we introduced. Both the corrected annotations and the original dataset annotations were considered valid during evaluation.

G LLM as a Judge

We use the `gpt4o mini` model (OpenAI, 2023), chosen for its cost-effectiveness and reliable performance. For both LLMs, we designed one prompt per field type. These prompts were automatically generated for each field type using the following prompt templates for format checking (see Listing 4) and for fact checking (see Listing 5), requiring only minor adjustments afterward.

Listing 4: Meta-Prompt for Generating Factual Correctness Prompts

```
I have a corpus of invoices for political advertisements from TV channels,
where information should be extracted. The following set of information
should be extracted:
---
advertiser
agency
channel
contract_num
flight_from
flight_to
gross_amount
product
program_desc
program_end_date
program_start_date
property
sub_amount
tv_address
---
I want to verify the factual correctness of each of these information using
a prompt. Please generate a prompt for every information in the style of
this prompt for the tv-address and output them in the form of a Python dictionary:

factual_correctness_prompts = {
    "tv_address": "You will evaluate addresses to determine if they are
    likely to be the official locations of a TV channel or broadcasting
    company. For each address:

    Assess Suitability: Evaluate whether the address could realistically
    serve as a media or broadcasting location. Consider factors such as
    the presence of corporate offices, proximity to media hubs, or known
    broadcasting facilities that would support its use as a TV channel address.

    Provide a Confidence Score: Based on this assessment, assign a confidence
    score from 0 to 1, reflecting how likely it is that the string is a valid
    location for a TV channel.

    Output Format (CSV):
    score; justification
    <numerical score (0 to 1)>; '<Short explanation of the score, highlighting
    specific aspects of the address that support or detract from its
    completeness and correctness.'>"
}
```

Table 7: List of Documents with the field types and corrected values, that are added to the VRDU ground truth for evaluation

Document	Field Type	Field Value
b0ae0954-274a-f270-797c-76224b78b8ee.pdf	agency	Del Ray Media
89b8c007-4189-bfa6-e0a5-fe1d173edf92.pdf	flight_from	05/27/20
89b8c007-4189-bfa6-e0a5-fe1d173edf92.pdf	flight_to	05/31/20
42adf390-6e50-6fbc-fbbe-65117a1ffcb2.pdf	gross_amount	$500.00
143af697-c6f9-a36e-d43f-1a92e800ffeb.pdf	flight_from	07/01/20
143af697-c6f9-a36e-d43f-1a92e800ffeb.pdf	flight_to	07/07/20
14632210-11d9-a184-e3db-b1b219f52ca8.pdf	flight_from	06/02/20
14632210-11d9-a184-e3db-b1b219f52ca8.pdf	flight_to	06/08/20
48845b9d-9e1b-a9e8-d560-58d35d2b31b2.pdf	flight_from	01/01/20
48845b9d-9e1b-a9e8-d560-58d35d2b31b2.pdf	flight_to	01/08/20
45f3875f-2b24-42fe-ddb4-fa203f4eec30.pdf	flight_from	01/22/20
45f3875f-2b24-42fe-ddb4-fa203f4eec30.pdf	flight_to	02/05/20
4cc700a3-6cb8-b791-2428-890e7fb7cf2a.pdf	flight_from	10/06/20
4cc700a3-6cb8-b791-2428-890e7fb7cf2a.pdf	flight_to	10/12/20
64243566-745a-3edd-224b-542129a844a6.pdf	flight_from	Apr16/20
64243566-745a-3edd-224b-542129a844a6.pdf	flight_to	Apr22/20
64243566-745a-3edd-224b-542129a844a6.pdf	product	POLITICIAL
7b9c8208-d2be-2a81-8a15-215a9a5a26e8.pdf	flight_from	Feb15/20
7b9c8208-d2be-2a81-8a15-215a9a5a26e8.pdf	flight_to	Feb21/20
7b9c8208-d2be-2a81-8a15-215a9a5a26e8.pdf	product	BLOOMBERG 4 PRES
dbd4ed2d-11f1-ba35-cc98-43127897504a.pdf	flight_from	Jun05/20
dbd4ed2d-11f1-ba35-cc98-43127897504a.pdf	flight_to	Jun19/20
dbd4ed2d-11f1-ba35-cc98-43127897504a.pdf	product	OWENS FOR CON UT04
0be55a7b-c4b9-7956-d523-30f79a4ebc1a.pdf	flight_from	1/27/2020
0be55a7b-c4b9-7956-d523-30f79a4ebc1a.pdf	flight_to	2/23/2020
4b330586-f3e8-28ea-b0cc-2d060dc10622.pdf	flight_from	4/27/2020
4b330586-f3e8-28ea-b0cc-2d060dc10622.pdf	flight_to	5/31/2020
38a1ec3a-18bd-0b73-1155-b6ced503f7a1.pdf	flight_from	1/27/2020
38a1ec3a-18bd-0b73-1155-b6ced503f7a1.pdf	flight_to	2/23/2020
c1ede720-d1f9-dcb4-e56f-65bf46300e84.pdf	flight_from	02/11/20
c1ede720-d1f9-dcb4-e56f-65bf46300e84.pdf	flight_to	02/17/20
cda5811d-3cf3-9c50-0941-28094bf9880f.pdf	flight_from	01/01/20
cda5811d-3cf3-9c50-0941-28094bf9880f.pdf	flight_to	01/08/20
b009ea0d-d54e-7410-320d-2dc99dbc8c09.pdf	tv_address	PO BOX 206270 Dallas, TX 75320-6270
b009ea0d-d54e-7410-320d-2dc99dbc8c09.pdf	flight_from	2/1/2020
b009ea0d-d54e-7410-320d-2dc99dbc8c09.pdf	flight_to	2/29/2020
a5a37afc-bbf5-db26-bd19-a71fee1ae67a.pdf	flight_from	05/06/20
88fa6e33-408f-6ac2-a253-d30e32bce302.pdf	flight_from	03/31/20
88fa6e33-408f-6ac2-a253-d30e32bce302.pdf	flight_to	04/05/20
80ff3aa4-3617-496e-fc29-cf9fdbecc54d.pdf	flight_from	04/29/20
80ff3aa4-3617-496e-fc29-cf9fdbecc54d.pdf	flight_to	05/05/20
65ebbb18-8a01-357a-94ce-bfa16723822e.pdf	flight_from	06/09/20
65ebbb18-8a01-357a-94ce-bfa16723822e.pdf	flight_to	06/15/20
64780ed0-180c-a77b-bfe0-478c2a48a3a0.pdf	flight_from	05/19/20
64780ed0-180c-a77b-bfe0-478c2a48a3a0.pdf	flight_to	05/22/20
73badb45-b62c-0c2e-1a2f-4b5fb4fda5b9.pdf	tv_address	6301 Bandel Road NW ROCHESTER Rochester, MN 55901-8798
73badb45-b62c-0c2e-1a2f-4b5fb4fda5b9.pdf	flight_from	10/06/20
73badb45-b62c-0c2e-1a2f-4b5fb4fda5b9.pdf	flight_to	10/12/20
685a2568-66ec-e3e0-27a1-78b56402b8cb.pdf	flight_from	09/15/20
685a2568-66ec-e3e0-27a1-78b56402b8cb.pdf	flight_to	09/21/20
2437f486-c8f2-72ad-1c75-4d45eedd57d2.pdf	flight_from	05/05/20
2437f486-c8f2-72ad-1c75-4d45eedd57d2.pdf	flight_to	05/05/20
c1f3f40f-9003-6d17-9d92-f7d62836f017.pdf	flight_from	04/06/20
c1f3f40f-9003-6d17-9d92-f7d62836f017.pdf	flight_to	04/13/20
35b3207f-bd16-d173-09a5-570acac710b2.pdf	flight_from	03/30/20
35b3207f-bd16-d173-09a5-570acac710b2.pdf	flight_to	04/06/20

Listing 5: Meta-Prompt for Generating Factual Correctness Prompts with JSON Output

```
I have a corpus of invoices for political advertisements from TV channels,
where information should be extracted. The following set of information
should be extracted:
---
advertiser
agency
channel
contract_num
flight_from
flight_to
gross_amount
product
program_desc
program_end_date
program_start_date
property
sub_amount
tv_address
---
I want to verify the factual correctness of each of these information using
a prompt. Please generate a prompt for every information in the style of
this prompt for the tv-address and add it to this JSON file:

factual_correctness_prompts = {
    "tv_address": "You will evaluate addresses to determine if they are
    likely to be the official locations of a TV channel or broadcasting
    company. For each address:

    Assess Suitability: Evaluate whether the address could realistically
    serve as a media or broadcasting location. Consider factors such as
    the presence of corporate offices, proximity to media hubs, or known
    broadcasting facilities that would support its use as a TV channel address.

    Provide a Confidence Score: Based on this assessment, assign a confidence
    score from 0 to 1, reflecting how likely it is that the string is a valid
    location for a TV channel.

    Output Format (CSV):
    score; justification
    <numerical score (0 to 1)>; '<Short explanation of the score, highlighting
    specific aspects of the address that support or detract from its
    completeness and correctness.'>"
}
```