International Journal of Applied Information Systems (IJAIS) – ISSN : 2249-0868
Foundation of Computer Science FCS, New York, USA
Volume 13– No.5, November 2026 – www.ijais.org

IntelliDocFormer: An Extended LayoutLMv3 Architecture with Graph-based Relational Reasoning and Document-Type-Aware Multimodal Fusion

Gabriel Ihuoma Lilian
Department of Computer Science
University of Port Harcourt
Port Harcourt, Nigeria

Laud Charles Ochei
Department of Computer Science
University of Port Harcourt
Port Harcourt, Nigeria

Uche Oghenekaro
Department of Computer Science
University of Port Harcourt
Port Harcourt, Nigeria

ABSTRACT

The digitisation of finance, healthcare, legal services and academia has produced large volumes of unstructured and semi-structured documents, from invoices and forms to scientific papers and contracts. Extracting information from them requires systems that understand both textual content and visual layout. Transformer-based models such as LayoutLM, DocFormer and Donut process documents multimodally, but they reason weakly about relations between spatially separated yet semantically related elements, treat all document types uniformly, have used visual encoders designed for natural scenes, and are computationally demanding. This paper presents IntelliDocFormer, an extended LayoutLMv3-based architecture that adds a Document-Type Classification (DTC) module, graph-based relational reasoning with a Graph Attention Network over a sparse layout graph, a Vision Transformer with Feature Pyramid Network (ViT-FPN) visual encoder, and an adaptive multimodal fusion layer. The architecture is analysed against existing models and evaluated with the Architecture Trade-off Analysis Method (ATAM) using a résumé-screening scenario extended to a utility tree of seven quality-attribute scenarios covering accuracy, throughput, degraded input, long documents, modifiability, explainability and fairness. The analysis shows that the modular design addresses document heterogeneity and complex layouts, and it identifies the sensitivity points, trade-offs and risks of the design, including the retained quadratic cost of the backbone. The contribution is conceptual and architectural; an implementation and validation plan on DocBank, FUNSD, CORD and DocVQA is specified for future empirical work.

General Terms

Document Intelligence, Deep Learning, Software Architecture, Information Extraction.

Keywords

Document parsing, Data structuring, Transformer architecture, LayoutLMv3, Multimodal fusion, Graph neural networks, Document-type classification, ATAM.

1. INTRODUCTION

Organisations generate vast quantities of documents, such as invoices, application forms, research papers, contracts and medical records, whose valuable data are embedded in diverse formats and layouts. Rule-based and template-driven approaches lack the flexibility to adapt to new layouts and content variations, which limits their effectiveness in dynamic environments [1]. Transformer-based architectures have changed document understanding by fusing textual, visual and spatial layout information [2]. LayoutLM [2], DocFormer [1] and Donut [3] perform well on entity recognition, key-value extraction and document classification, yet several limitations persist:

(1) **Limited relational reasoning:** sequential tokenisation with 2-D position embeddings struggles to capture dependencies between spatially separated but related elements, such as form labels and their values [4, 5].
(2) **No document-type awareness:** most models apply the same processing to all categories (e.g. invoice, résumé, scientific paper), although these follow distinct structural conventions.
(3) **Visual encoding:** early models obtain visual features from object detectors such as Faster R-CNN [6], designed for natural scenes rather than document layouts [2].
(4) **Computational cost:** self-attention scales quadratically with input length, $O(n^2 d)$, making long documents expensive [7, 8].
(5) **Limited adaptability:** models tuned to specific document types or datasets transfer poorly to heterogeneous formats [1].

These limitations sit within broader challenges: heterogeneous formats, non-linear layouts (multi-column pages, nested tables, overlapping graphics), long-range context, alignment of OCR text, layout and visual features, and computational constraints [4, 8]. To address them, this paper presents IntelliDocFormer, an extended LayoutLMv3-based architecture for multimodal document parsing and data structuring. Its contributions are:

(1) An architectural analysis of transformer-based document-understanding models, their design patterns and limitations.
(2) The IntelliDocFormer architecture, which extends the LayoutLMv3 backbone with a DTC module, layout-graph reasoning with Graph Attention Networks (GATs), a ViT-FPN visual encoder and an adaptive multimodal fusion layer.
(3) A scenario-based evaluation using ATAM, extended to seven quality-attribute scenarios, identifying risks, trade-offs and the impact of changes on performance criteria.
(4) A design rationale and an implementation plan for empirical validation.

This is a conceptual architectural contribution: it proposes and analyses a design rather than reporting results from a trained model. Section 2 reviews related concepts and architectures, Section 3 analyses the existing system, Section 4 presents IntelliDocFormer, Section 5 reports the scenario-based evaluation, Section 6 gives the implementation plan, Section 7 discusses implications and Section 8 concludes.

2. RELATED CONCEPTS AND ARCHITECTURES

2.1 Document parsing and modalities

Document parsing extracts structured information from unstructured or semi-structured documents and converts it into machine-interpretable forms such as relational tables, JSON objects or knowledge graphs [2, 4]. Its core tasks are text extraction with Optical Character Recognition (OCR), which returns tokens and their bounding boxes [9]; layout analysis, which segments titles, paragraphs, tables and figures and determines reading order; information extraction of entities, key-value pairs and relations; and data structuring into standard formats [1]. Four modalities are involved: the *textual* modality, processed with contextual embeddings such as BERT [10]; the *layout* modality, usually encoded with 2-D position embeddings of bounding boxes; the *visual* modality of scanned pages, logos, fonts and graphics; and the *structural* modality of section hierarchies, lists, table schemas and cross-references.

2.2 Transformer-based document architectures

LayoutLM series. LayoutLM [2] extends BERT with 2-D position embeddings of token bounding boxes and, for fine-tuning, image features of token regions from Faster R-CNN; it is pre-trained with Masked Visual-Language Modelling (MVLM) and Multi-label Document Classification. LayoutLMv2 [11] adds image embeddings to pre-training, spatial-aware self-attention and text-image alignment and matching objectives. LayoutLMv3 [4] uses linear patch embeddings instead of a CNN backbone, unifies text and image masking and adds word-patch alignment, improving both text-centric and image-centric tasks. The series depends on OCR quality and does not model relations beyond positional proximity explicitly.

DocFormer [1] is an end-to-end multimodal transformer that combines text, visual and spatial features through multimodal self-attention with shared spatial embeddings; it needs large-scale training data and much memory, and remains OCR-dependent. **Donut** [3] is OCR-free: a Vision Transformer encoder and text decoder map images directly to structured output, but performance can lag on highly structured tabular data and extensive pre-training is needed.

TILT [12] is an encoder-decoder transformer in which layout is an attention bias, built on a T5-style model. **FormNet** [5] builds "Super-Tokens" that embed neighbour information through graph convolutions combined with Rich Attention over spatial relations; it is the closest prior work to the graph reasoning in IntelliDocFormer, although its graph convolution operates inside attention rather than as a separate GNN stage feeding a fusion layer. **StructuralLM** [13] treats each form cell as a semantic unit with a cell-position objective, whereas IntelliDocFormer uses token-level graph nodes. Table 1 compares these architectures.

Table 1. Comparison of existing document-understanding architectures

Model	Paradigm	Modalities	Key innovation	Limitation
LayoutLM [2]	BERT encoder + 2-D positions	Text, layout (+image)	Joint text-layout pre-training	OCR-dependent; limited visual use
LayoutLMv2 [11]	Multimodal transformer	Text, layout, image	Spatial-aware attention; text-image alignment	Costly; OCR-dependent
LayoutLMv3 [4]	Unified text-image masking	Text, layout, image	Word-patch alignment; patch embeddings	High compute; no explicit relations
DocFormer [1]	Joint multimodal encoder	Text, layout, image	Multimodal self-attention	Large data and memory needs
Donut [3]	Vision encoder + decoder	Image only	OCR-free, end-to-end	Weaker on structured tables
TILT [12]	Encoder-decoder	Text, layout, image	Layout as attention bias	High compute
FormNet [5]	Graph-enhanced transformer	Text, layout	Super-Tokens; Rich Attention	Form-like documents
StructuralLM [13]	LayoutLM extension	Text, layout, cells	Cell-level pre-training	Less suited to free text

2.3 Design patterns in existing architectures

Five recurring patterns were identified (Table 2). *Token and layout embedding* captures what is written and where, at the cost of OCR dependence [4]. *Multimodal fusion* gives holistic understanding but adds complexity [1, 2]. *End-to-end processing* removes OCR error propagation but needs larger datasets [3]. *Hierarchical modelling* captures document organisation at the cost of depth [13]. *Graph-based and unified pre-training* is the pattern most directly motivating IntelliDocFormer: Doc2Graph [14] builds a document graph whose nodes are text regions and whose edges encode spatial and logical relations, processed by a GNN, while UDOP [15] unifies vision, text and layout in one sequence-to-sequence model. DocLLM [16] reasons over text and layout in a layout-aware generative language model without a graph or visual encoder. A further trend applies large vision-language models directly to document images without OCR or graph construction, as in KOSMOS-2.5 [17], mPLUG-DocOwl [18] and Nougat [19]. These models trade the explicit, interpretable pipeline of IntelliDocFormer (OCR, typed embeddings, graph, fusion) for generative flexibility; whether explicit relational structure or implicit structure learned in a large model is preferable remains open and is a boundary condition on this contribution, not a claim of superiority.

Table 2. Design patterns in existing architectures

Pattern	Examples	Strength	Weakness
Token + layout embedding	LayoutLM, DocFormer	Text and position	OCR dependence
Multimodal fusion	LayoutLMv2, DocFormer, TILT	Holistic view	Compute overhead
End-to-end	Donut, Nougat	No OCR errors	Data hungry
Hierarchical	StructuralLM	Organisation	Depth, training time
Graph-based	Doc2Graph, FormNet	Explicit relations	Graph design choices

3. ANALYSIS OF THE EXISTING SYSTEM

3.1 Existing system architecture

LayoutLM [2] is analysed as the baseline because of its pioneering role and wide adoption (Figure 1). In the **input layer**,

a scanned document passes through OCR (e.g. Tesseract [9]), which returns tokens and bounding boxes (x_0, y_0, x_1, y_1). In the **embedding layer**, each WordPiece token receives a text embedding, a 1-D position embedding and 2-D position embeddings of its box, discretised to integers in the range 0–1000; visual features of token regions are extracted with Faster R-CNN [6]. A stack of **transformer encoder** layers, pre-trained with MVLM and multi-label document classification, produces contextual token representations, and the **output layer** applies token-level or document-level ([CLS]) classification heads.

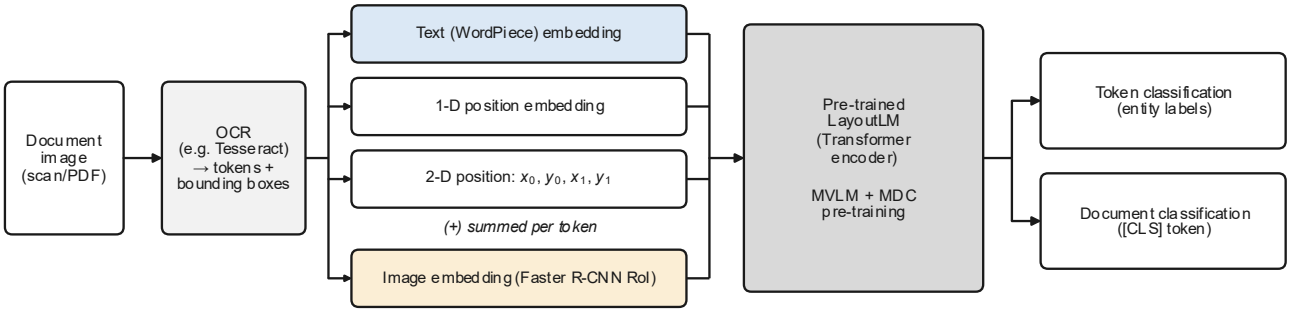


Fig 1: Architecture of the existing LayoutLM-based system (adapted from Xu et al. [2])

The baseline algorithm is: (1) input document D; (2) perform OCR if D is an image or PDF; (3) tokenise D; (4) build text, layout and visual embeddings; (5) encode with the multimodal transformer; (6) apply a token- or document-classification head; and (7) structure the extracted data into the desired format.

3.2 Advantages and disadvantages

LayoutLM achieves high accuracy on documents with consistent formats: it raised the F1 score for form understanding on FUNSD from 70.72 to 79.27 and for receipt understanding on SROIE from 94.02 to 95.24 [2]. OCR and transformer pipelines are widely adopted in finance, healthcare, legal services and retail, and open tools such as Tesseract, EasyOCR and PaddleOCR are readily available. On the other hand, such models falter on layouts that differ from the training data [2], underperform on weakly structured documents such as contracts and multi-column papers [20], need manual annotation and fine-tuning for each new document type, may miss domain-specific semantics, depend heavily on OCR quality, are limited to 512 tokens, and are computationally demanding.

3.3 Performance gaps

Six gaps follow from the analysis: weak modelling of relations between distant but related elements; no adaptation to document category; visual features from detectors designed for natural scenes; no explicit document graph for relation extraction; quadratic attention cost; and limited joint modelling of text, layout and visual features. Figure 2 pairs each gap with the corresponding architectural response in IntelliDocFormer.

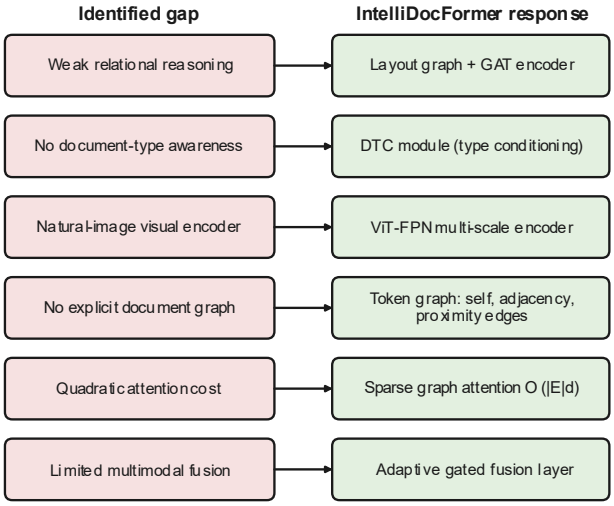


Fig 2: Identified gaps and IntelliDocFormer responses

4. PROPOSED ARCHITECTURE: INTELLIDOCFORMER

4.1 Design philosophy

IntelliDocFormer aims to be a robust and scalable framework for multimodal document understanding that merges text, layout and image into a unified representation [1] and adapts to document structures found in legal, medical and financial processing. Its guiding principles are comprehensive multimodal integration, graph-based relational reasoning, document-type awareness, enhanced visual encoding, computational efficiency, and modularity and extensibility.

4.2 High-level architecture and data flow

Figure 3 shows the layered architecture and Table 3 summarises its main components.

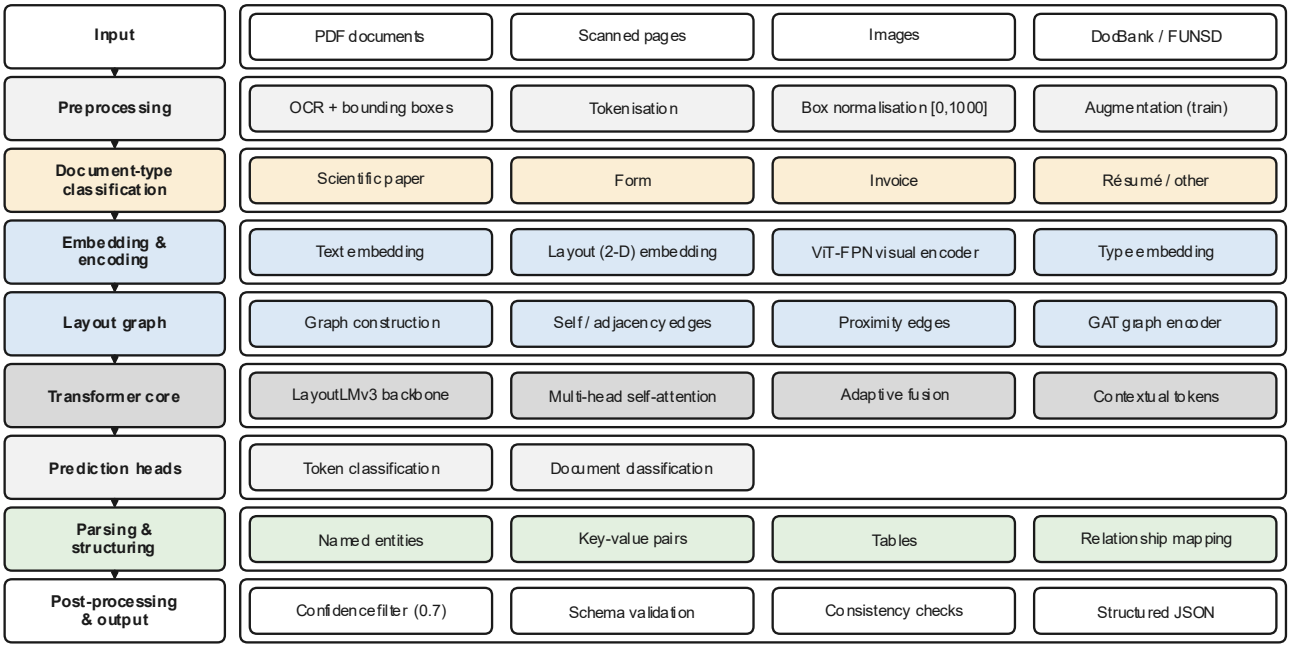


Fig 3: Layered architecture of IntelliDocFormer

Table 3. Main architectural components

Component	Function	Key innovation
Preprocessing	OCR, tokenisation, box normalisation, augmentation	Multi-stage pipeline
DTC module	Early document-category prediction	Type-conditioned processing
Embedding layer	Text, layout, visual and type embeddings	ViT-FPN visual encoding
Layout graph	Graph construction and GAT reasoning	Explicit relational modelling
Transformer core	LayoutLMv3 backbone and fusion	Graph- and type-aware fusion
Parsing engine	NER, classification, tables, key-value pairs, relations	Graph-based relation extraction
Post-processing	Formatting, validation, error handling	Confidence filtering; schema checks

The data flow (Figure 4) proceeds as follows: the document is ingested in its native form; the DTC module predicts its category; OCR extracts text and boxes and the text is tokenised; text, layout, visual and graph encoders run in parallel; the fusion layer integrates their outputs, conditioned on the document type; and token- and document-classification heads produce the output, which is post-processed into structured JSON.

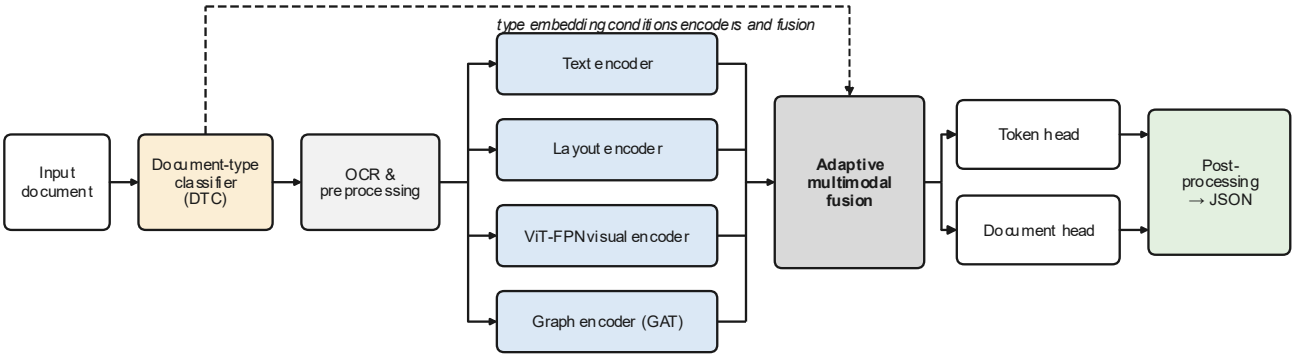


Fig 4: Data flow of the proposed system

4.3 Preprocessing and embedding

For scanned documents, OCR maps every token to a bounding box; for corpora with existing annotations, such as DocBank [21], the LayoutLMv3 processor is used with OCR disabled (apply_ocr = False). Text is tokenised with the byte-level BPE tokenizer of LayoutLMv3, which handles unknown words through sub-word units. Box coordinates are normalised to the range 0–1000 relative to the page width W and height H:

$$\hat{x} = \lfloor 1000\, x/W \rfloor, \qquad \hat{y} = \lfloor 1000\, y/H \rfloor \tag{1}$$

During training, augmentation with colour jitter (brightness, contrast and saturation 0.2, hue 0.05), random rotation (3°) and Gaussian noise (standard deviation 6.0) improves robustness. Each token then receives text, 1-D position and segment embeddings, and layout embeddings of $(\hat{x}_0, \hat{y}_0, \hat{x}_1, \hat{y}_1)$; visual embeddings come from the ViT-FPN encoder (Section 4.6).

4.4 Document-type classification module

Existing models apply an identical extraction strategy to every document category, neglecting the structural and functional differences between, for example, invoices and résumés.

IntelliDocFormer places a lightweight DTC module at the start of the pipeline. It predicts a distribution $\mathbf{p}$ over K categories from a pooled visual feature $\mathbf{v}$, forms a type vector $\mathbf{t}$ from a learned type-embedding matrix E_{type}, and adds it to each token embedding $\mathbf{e}_i$:

$$\mathbf{p} = \text{softmax}(W_c\,\mathbf{v} + \mathbf{b}_c) \tag{2}$$

$$\mathbf{t} = E_{type}^{\top}\,\mathbf{p} \tag{3}$$

$$\mathbf{e}_i{}' = \mathbf{e}_i + \mathbf{t} \tag{4}$$

Using the soft distribution rather than a hard label keeps the module differentiable and limits the damage of misclassification. The conditioning vector is expected to let attention and fusion focus on regions that matter for each type, supporting key-value extraction, entity recognition and layout parsing.

4.5 Layout graph embeddings with GNNs

The document is represented as a graph whose nodes are tokens with their boxes (Figure 5). Three explicit edge types are used: self-edges, sequential adjacency edges between consecutive tokens, and spatial-proximity edges between tokens whose box centres $\mathbf{c}_i$ and $\mathbf{c}_j$ lie within a threshold distance δ:

$$\mathcal{E} = \mathcal{E}_{self} \cup \mathcal{E}_{adj} \cup \mathcal{E}_{prox} \tag{5}$$

where $\mathcal{E}_{self} = \{(i,i)\}$, $\mathcal{E}_{adj} = \{(i, i \pm 1)\}$ and $\mathcal{E}_{prox} = \{(i,j) : \| \mathbf{c}_i - \mathbf{c}_j \|_2 < \delta\}$.

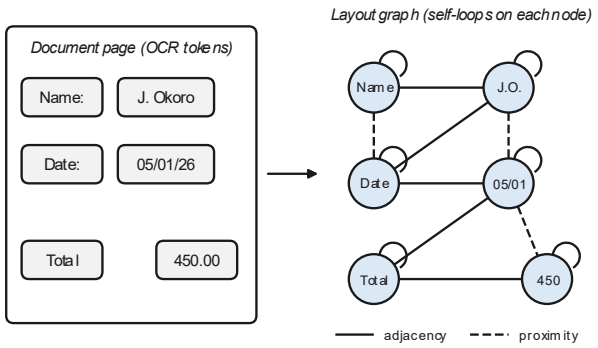


Fig 5: Transformation of a document into a layout graph

The graph encoder uses graph attention (GATConv) [22]. For node features $\mathbf{h}_i$, a shared weight matrix W, an attention vector $\mathbf{a}$ and neighbourhood $\mathcal{N}(i)$:

$$e_{ij} = \text{LeakyReLU}\big(\mathbf{a}^{\top}[W\mathbf{h}_i \,\|\, W\mathbf{h}_j]\big) \tag{6}$$

$$\alpha_{ij} = \frac{\exp(e_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(e_{ik})} \tag{7}$$

$$\mathbf{h}_i{}' = \sigma\big(\textstyle\sum_{j \in \mathcal{N}(i)} \alpha_{ij}\, W\mathbf{h}_j\big) \tag{8}$$

This construction differs from Doc2Graph [14], which builds a fully connected graph over entities and lets the network learn relevant connections with node and edge classifiers. IntelliDocFormer deliberately uses a sparser heuristic graph: it gives up some representational flexibility for simplicity, modularity and lower cost. Which approach is more accurate is an empirical question left to future work.

Complexity. Self-attention in the LayoutLMv3 backbone costs $O(n^2 d)$ for n tokens of dimension d [8]. Graph attention attends only over neighbourhoods and costs $O(|\mathcal{E}|\,d)$; because the edge types are limited, $|\mathcal{E}|$ grows linearly with n for a fixed threshold and neighbourhood density. The backbone is retained, so the total cost is

$$C = O(n^2 d) + O(|\mathcal{E}|\,d) \tag{9}$$

that is, the graph layer adds a sub-quadratic term but does not remove the dominant quadratic one. Latency and memory will be measured in the planned implementation.

4.6 ViT-FPN visual encoder

LayoutLM relies on Faster R-CNN region features [6], built for natural scenes, and LayoutLMv3 uses single-scale linear patch embeddings [4]. Document elements such as headers, labels and fields can be far apart yet related, and they vary greatly in size. IntelliDocFormer therefore uses a Vision Transformer [23], which splits the page into N patches $\mathbf{x}_p^k$ (e.g. 16×16 pixels), projects them with E, adds position embeddings E_{pos} and applies L layers of multi-head self-attention (MSA) with layer normalisation (LN):

$$\mathbf{z}_0 = \big[\mathbf{x}_p^1 E; \mathbf{x}_p^2 E; \ldots; \mathbf{x}_p^N E\big] + E_{pos} \tag{10}$$

$$\mathbf{z}_\ell{}' = \text{MSA}\big(\text{LN}(\mathbf{z}_{\ell-1})\big) + \mathbf{z}_{\ell-1} \tag{11}$$

$$\mathbf{z}_\ell = \text{MLP}\big(\text{LN}(\mathbf{z}_\ell{}')\big) + \mathbf{z}_\ell{}' \tag{12}$$

A Feature Pyramid Network [24] then builds multi-scale maps through a top-down pathway with lateral connections (Figure 6), capturing fine details and broader layout simultaneously, which is useful for table parsing and layout segmentation.

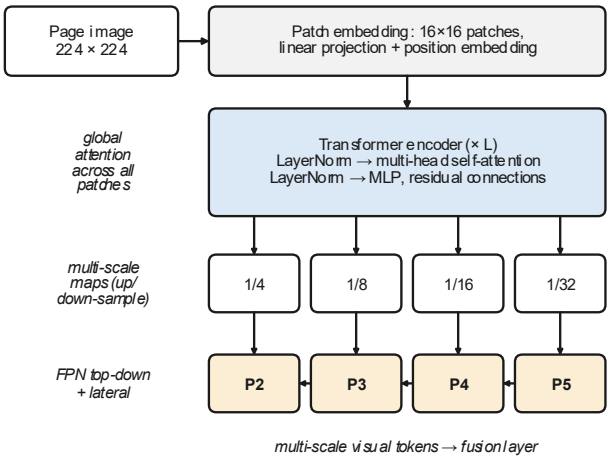


Fig 6: ViT-FPN visual encoder

4.7 Adaptive multimodal fusion

After graph processing, node representations H_g are fused with the LayoutLMv3 sequence output H_s (Figure 7). The fusion is a two-layer MLP with GELU activation [25] and dropout:

$$U = \text{GELU}\big(W_1[H_s; H_g] + \mathbf{b}_1\big) \tag{13}$$

$$F = W_2\,\text{Dropout}(U) + \mathbf{b}_2 \tag{14}$$

To weight modalities by task and document context, a gate computed from both representations and the type vector blends the fused and sequence representations:

$$\mathbf{g} = \sigma\big(W_g[H_s; H_g; \mathbf{t}] + \mathbf{b}_g\big) \tag{15}$$

$$\tilde{F} = \mathbf{g} \odot F + (1 - \mathbf{g}) \odot H_s \tag{16}$$

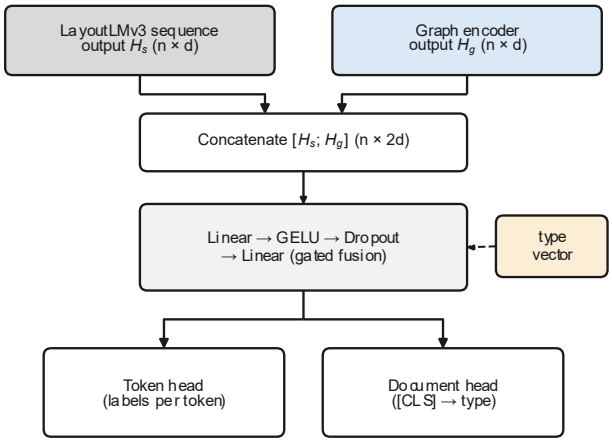


Fig 7: Fusion layer and task-specific heads

In form extraction, text and layout are expected to dominate; in image-centric tasks, the visual modality gains weight. The fused representation feeds a token-classification head and a document-classification head on the [CLS] token.

4.8 Parsing, structuring and post-processing

The parsing and structuring engine (Figure 8) has five components. Named Entity Recognition (NER) labels each token with a category such as title, author, abstract, section, paragraph, list, table, figure, equation, caption, footer or reference. Document classification assigns the document type from the [CLS] representation. Table extraction detects tables and their rows, columns and headers and converts them to JSON, CSV or relational form; PubTables-1M [26] is a natural resource for training and evaluating this module. Key-value extraction identifies keys and values from NER and proximity signals and pairs them with a classifier, an approach related to the graph-convolutional key-value linking of PICK [27], although IntelliDocFormer performs it downstream of its own graph encoder. Relationship mapping finally builds a document knowledge graph linking entities and values.

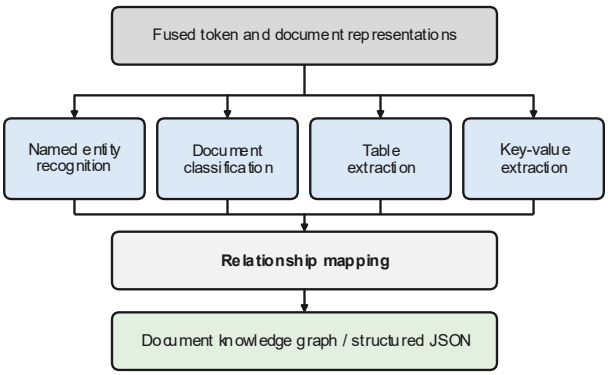


Fig 8: Parsing and structuring engine

Post-processing serialises the output to JSON with a document identifier, the predicted document type and, for each token, its text, label and box. Tokens whose confidence falls below 0.7 are flagged for review, outputs are validated against predefined schemas, and cross-field consistency checks (e.g. agreement between labels and boxes) are applied.

5. SCENARIO-BASED EVALUATION

5.1 Methodology

Architecture evaluation methods are commonly classified as experience-based, simulation-based, mathematical-modelling-based or scenario-based [28]. Scenario-based methods, such as the Software Architecture Analysis Method (SAAM) [29] and ATAM [30], suit architectures that have not been implemented: they describe interactions between the system and its environment and examine how the architecture meets functional and non-functional requirements [31]. This study applies ATAM, whose steps are preparation, evaluation meeting, identification of architectural approaches, trade-off analysis, and documentation and decision-making.

5.2 Primary scenario: job application processing

Description. A recruitment firm receives more than 1,000 résumés daily in varied formats (scanned, digital, tabular). Existing systems fail on unseen layouts and misalign key-value pairs, for example labelling "Experience: 5 years" as education. **Interaction.** A candidate submits a résumé, cover letter and optional certificates through a recruitment platform, which starts an automated screening pipeline. **IntelliDocFormer response.** The DTC module separates résumés, certificates and cover letters; layout-aware extraction associates section headers with content; graph reasoning captures relations between sections and their content; and validation cross-checks extracted fields against job requirements. **Evaluation criteria.** Extraction accuracy, false-positive and false-negative rates, computational efficiency, scalability and robustness to unusual inputs and new document types.

Fairness, explainability and regulation. Automated résumé screening raises fairness and transparency concerns. Disparate-impact testing across applicant subgroups and document formats is needed, a documented human-in-the-loop review must remain because the system assists rather than replaces hiring decisions, and explanations of extraction outputs must be available to reviewers and auditors. Under the EU AI Act, AI systems used in employment and recruitment are classified as high-risk [32], and the GDPR applies to the personal data processed, including its provisions on automated decision-making [33]. These concerns are represented as scenarios S6 and S7 below; their empirical assessment is future work.

5.3 Application of ATAM

The ATAM application (Table 4) was conducted by the authors as a structured walkthrough following the published ATAM steps. Formal workshops with developers, HR professionals and candidates were not held in this iteration; the analysis is a self-assessment against the ATAM framework, not externally elicited stakeholder feedback.

Table 4. Application of ATAM to the document-parsing scenario

ATAM step	Description	Application to the scenario
Preparation	Define stakeholders, goals, constraints and scenarios.	Stakeholders: developers, HR professionals, candidates. Goals: accurate extraction, near-real-time processing, scalability, robustness. Scenarios S1–S7 (Figure 9).
Evaluation meeting	Present design and scenarios; elicit concerns.	Self-assessment against likely priorities: developers, maintainability; HR, accuracy and turnaround; candidates, fair and accurate processing.

ATAM step	Description	Application to the scenario
Identify architectural approaches	Explore alternatives that address concerns.	Microservices, event-driven processing, distributed inference; sparse versus fully connected graphs; OCR-based versus OCR-free encoders.
Analyse trade-offs	Analyse effects on quality attributes.	Extraction accuracy versus computational efficiency; complexity and maintenance overhead; interpretability versus end-to-end flexibility (Table 5).
Documentation and decision-making	Record findings and recommendations.	Adopt the sparse heuristic graph and modular encoders; plan ablations of DTC, graph and ViT-FPN; iterate the design after empirical results.

5.4 Extended scenarios and utility tree

To evaluate the architecture across more conditions than the résumé case, the quality goals were refined into seven scenarios arranged as an ATAM utility tree (Figure 9) and ranked by importance and difficulty. They cover résumé key-value linking (S1), invoice processing at peak volume (S2), degraded scans (S3), long scientific articles (S4), addition of a new document type (S5), auditability (S6) and consistency across résumé templates (S7). Table 5 analyses each scenario.

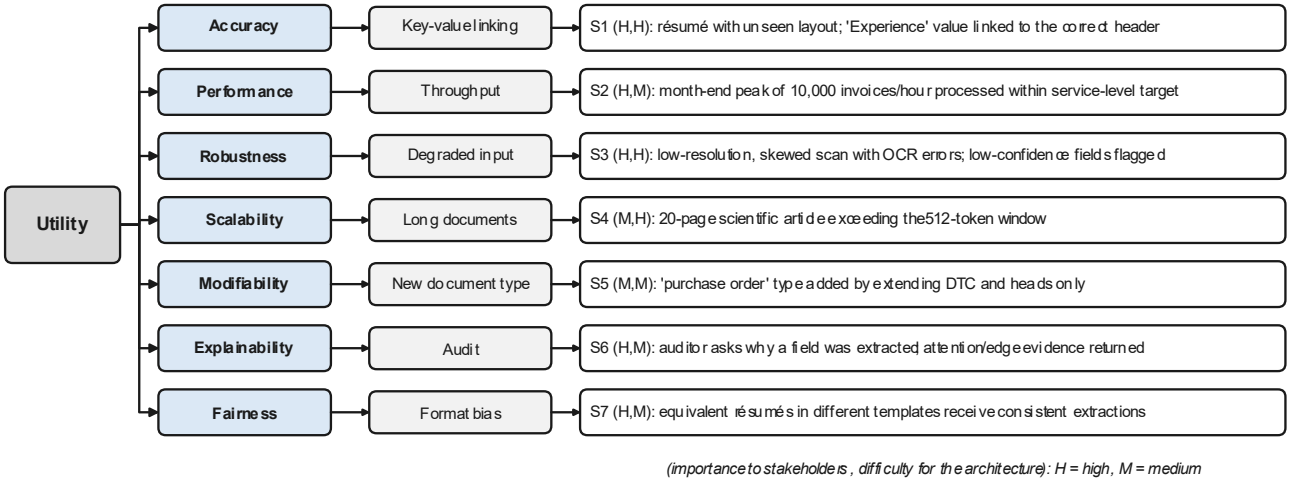


Fig 9: ATAM utility tree with seven quality-attribute scenarios

Table 5. Scenario analysis: decisions, sensitivity and trade-off points, risks and non-risks

Scenario	Supporting decisions	Sensitivity / trade-off points	Risk (R) / non-risk (N)
S1 Key-value linking	Proximity edges; GAT; key-value classifier	Threshold δ trades recall of distant pairs against graph size	N: relations modelled explicitly, unlike baseline
S2 Invoice peak	Stateless inference services; batching; DTC routing	Graph and ViT-FPN stages add latency per page	R: backbone keeps $O(n^2 d)$ cost (Eq. 9)
S3 Degraded scans	Augmentation; confidence filter; visual stream	Filter threshold trades manual review load against error rate	R: extraction still coupled to OCR quality
S4 Long article	Page-wise chunking; sparse graph across chunks	Chunk size trades context against memory	R: 512-token limit of backbone
S5 New type	Modular DTC and heads; type embeddings	Number of types affects DTC accuracy	N: backbone and encoders unchanged
S6 Audit	Attention weights α_{ij}; edge evidence; logs	Explanation detail trades against storage	R: attention is only partial explanation
S7 Template fairness	Layout-normalised boxes; human review	Type conditioning may encode template bias	R: disparate-impact tests not yet run

5.5 Risks, trade-offs and impact of changes

Overfitting (risk). Fusing several modalities with graph embeddings increases capacity and the risk of memorising training data; dropout, normalisation and augmentation mitigate it. **Accuracy versus efficiency (trade-off).** Multimodal fusion can improve parsing accuracy but slows inference; quantisation and efficient encoders address this. **OCR dependency (vulnerability).** Inaccurate OCR misleads the model, so validation and preprocessing of OCR output are essential.

The expected impact of architectural changes is as follows. Multimodal fusion with graph embeddings is expected to improve detection accuracy and reduce false positives and negatives by giving a more complete view of the input. Pruning and hardware acceleration are expected to improve computational efficiency. Distributed computing is expected to improve scalability. Diverse modalities are expected to make the model harder to manipulate, improving robustness. These are design expectations, not measured outcomes.

6. IMPLEMENTATION AND VALIDATION PLAN

6.1 Datasets

Table 6 lists the datasets selected for the empirical study. DocBank and FUNSD are the primary training and evaluation sets; no training has yet been conducted on them. PubLayNet [34], with about 360,000 layout-annotated scholarly pages, and RVL-CDIP [35], with 400,000 images in 16 classes, are candidates for extending the evaluation, the latter being particularly suited to the DTC module. The preprocessing pipeline comprises OCR with Tesseract for scanned pages, tokenisation and box normalisation, alignment of text, layout and visual inputs, and augmentation.

Table 6. Datasets for empirical validation

Dataset	Content	Purpose
DocBank [21]	500,000 pages with token-level layout labels	Token-level parsing, layout analysis
FUNSD [36]	199 annotated forms with entity links	Form understanding, key-value linking
CORD [37]	1,000 public annotated receipts	Semi-structured extraction
DocVQA [38]	12,000+ document images with QA pairs	Document-level comprehension

6.2 Training configuration and metrics

Table 7 gives the planned training configuration, with separate learning rates for the pre-trained encoder and the new heads.

Table 7. Planned training hyperparameters

Parameter	Value
Backbone	microsoft/layoutlmv3-base
Image size	224 × 224
Maximum sequence length	512
Batch size (train / validation)	4 / 4
Epochs	30
Encoder learning rate	1×10^{-5}
Head learning rate	5×10^{-5}

Extraction is evaluated with precision, recall and F1-score from true positives (TP), false positives (FP) and false negatives (FN), and layout detection with Intersection over Union (IoU) between predicted and reference regions A and B:

$$P = \frac{TP}{TP+FP}, \qquad R = \frac{TP}{TP+FN} \tag{17}$$

$$F_1 = \frac{2PR}{P+R}, \qquad \text{IoU} = \frac{|A \cap B|}{|A \cup B|} \tag{18}$$

Efficiency is measured as inference latency (seconds per sample) and throughput (samples per second). Ablation studies will remove the DTC module, the graph encoder and the ViT-FPN encoder in turn to measure the contribution of each.

6.3 Technology stack and baselines

The implementation uses Python with PyTorch [39], HuggingFace Transformers [40] for the LayoutLMv3 backbone and processor, PyTorch Geometric [41] for graph layers, scikit-learn for metrics, Matplotlib and Seaborn for visualisation, and Docker for reproducible deployment. IntelliDocFormer will be compared with LayoutLMv3 (token and document classification), DocFormer (entity extraction and form understanding) and Donut (image-to-structured output) on the same tasks.

7. DISCUSSION

Robustness and multimodal integration. The architecture is designed to parse documents of varied structural complexity and to generalise across heterogeneous layouts without extensive retraining. Its cross-modal fusion is intended to align text and visual features more tightly, consistent with the rationale of prior multimodal work; this expectation motivates the design rather than reporting a measured outcome.

Modifiability and reusability. Separate encoders for text, layout, visual and graph inputs can be improved or replaced independently; for example, a better OCR engine or layout encoder can be integrated with little change elsewhere, which eases adaptation to new domains (scenario S5).

Limitations and trade-offs. Performance is expected to degrade on low-resolution, skewed or unconventional documents, reflecting a trade-off between graph-based structural modelling and robustness to visual distortion. The graph and fusion components add computational overhead, and the retained backbone keeps the quadratic attention cost. Extraction remains coupled to OCR quality, and large documents may exceed memory limits, requiring chunking or hierarchical strategies.

Implications. Combining visual, textual, layout and relational cues is promising for high-stakes domains such as healthcare, finance and legal services, and the modular design supports both scaling and domain-specific customisation, provided that fairness, explainability and regulatory requirements are addressed in deployment.

8. CONCLUSION

This paper presented IntelliDocFormer, an extended LayoutLMv3-based architecture for multimodal document parsing and data structuring. It contributes an architectural analysis of existing models; an extended framework with a Document-Type Classification module, layout-graph embeddings processed by Graph Attention Networks, a ViT-FPN visual encoder and adaptive multimodal fusion; a scenario-based ATAM evaluation extended to seven quality-attribute scenarios; and an implementation plan. Its distinguishing feature is type-aware, graph-informed cross-modal reasoning, in contrast to the uniform processing of several existing systems, which makes it a candidate for high-precision document automation in legal, healthcare and financial services.

The main limitation is the conceptual nature of the contribution: training performance, deployment behaviour and benchmark results remain to be established. Future work will prototype the model, run ablation studies and evaluate it on DocBank, FUNSD, CORD and DocVQA against the baselines in Section 6.3, including latency, memory and fairness measurements.

9. REFERENCES

[1] Appalaraju, S., Jasani, B., Kota, B.U., Xie, Y. and Manmatha, R. 2021 DocFormer: End-to-end transformer for document understanding. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV). 993-1003.

[2] Xu, Y., Li, M., Cui, L., Huang, S., Wei, F. and Zhou, M. 2020 LayoutLM: Pre-training of text and layout for document image understanding. Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining. 1192-1200.

[3] Kim, G., Hong, T., Yim, M., Nam, J., Park, J., Yim, J., Hwang, W., Yun, S., Han, D. and Park, S. 2022 OCR-free

document understanding transformer. Proceedings of the European Conference on Computer Vision (ECCV). 498-517.

[4] Huang, Y., Lv, T., Cui, L., Lu, Y. and Wei, F. 2022 LayoutLMv3: Pre-training for document AI with unified text and image masking. Proceedings of the 30th ACM International Conference on Multimedia. 4083-4091.

[5] Lee, C.-Y., Li, C.-L., Dozat, T., Perot, V., Su, G., Hua, N., Ainslie, J., Wang, R., Fujii, Y. and Pfister, T. 2022 FormNet: Structural encoding beyond sequential modeling in form document information extraction. Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (ACL). 3735-3754.

[6] Ren, S., He, K., Girshick, R. and Sun, J. 2015 Faster R-CNN: Towards real-time object detection with region proposal networks. Advances in Neural Information Processing Systems (NeurIPS). 28: 91-99.

[7] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, Ł. and Polosukhin, I. 2017 Attention is all you need. Advances in Neural Information Processing Systems (NeurIPS). 30: 5998-6008.

[8] Beltagy, I., Peters, M.E. and Cohan, A. 2020 Longformer: The long-document transformer. arXiv preprint arXiv:2004.05150.

[9] Smith, R. 2007 An overview of the Tesseract OCR engine. Proceedings of the Ninth International Conference on Document Analysis and Recognition (ICDAR). 629-633.

[10] Devlin, J., Chang, M.-W., Lee, K. and Toutanova, K. 2019 BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT 2019. 4171-4186.

[11] Xu, Y., Xu, Y., Lv, T., Cui, L., Wei, F., Wang, G., Lu, Y., Florencio, D., Zhang, C., Che, W., Zhang, M. and Zhou, L. 2021 LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. Proceedings of ACL-IJCNLP 2021. 2579-2591.

[12] Powalski, R., Borchmann, Ł., Jurkiewicz, D., Dwojak, T., Pietruszka, M. and Pałka, G. 2021 Going full-TILT boogie on document understanding with text-image-layout transformer. Proceedings of the International Conference on Document Analysis and Recognition (ICDAR). 732-747.

[13] Li, C., Bi, B., Yan, M., Wang, W., Huang, S., Huang, F. and Si, L. 2021 StructuralLM: Structural pre-training for form understanding. Proceedings of ACL-IJCNLP 2021. 6309-6318.

[14] Gemelli, A., Biswas, S., Civitelli, E., Lladós, J. and Marinai, S. 2022 Doc2Graph: A task agnostic document understanding framework based on graph neural networks. Computer Vision – ECCV 2022 Workshops. Springer. 329-344.

[15] Tang, Z., Yang, Z., Wang, G., Fang, Y., Liu, Y., Zhu, C., Zeng, M., Zhang, C. and Bansal, M. 2023 Unifying vision, text, and layout for universal document processing. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 19254-19264.

[16] Wang, D., Raman, N., Sibue, M., Ma, Z., Babkin, P., Kaur, S., Pei, Y., Nourbakhsh, A. and Liu, X. 2024 DocLLM: A layout-aware generative language model for multimodal document understanding. Proceedings of the 62nd Annual Meeting of the Association for Computational Linguistics (ACL). 8529-8548.

[17] Lv, T., Huang, Y., Chen, J., Cui, L., Ma, S., Chang, Y., Huang, S., Wang, W., Dong, L., Luo, W. et al. 2023 Kosmos-2.5: A multimodal literate model. arXiv preprint arXiv:2309.11419.

[18] Ye, J., Hu, A., Xu, H., Ye, Q., Yan, M. et al. 2023 mPLUG-DocOwl: Modularized multimodal large language model for document understanding. arXiv preprint arXiv:2307.02499.

[19] Blecher, L., Cucurull, G., Scialom, T. and Stojnic, R. 2023 Nougat: Neural optical understanding for academic documents. arXiv preprint arXiv:2308.13418.

[20] Chalkidis, I., Fergadiotis, M., Malakasiotis, P., Aletras, N. and Androutsopoulos, I. 2020 LEGAL-BERT: The muppets straight out of law school. Findings of the Association for Computational Linguistics: EMNLP 2020. 2898-2904.

[21] Li, M., Xu, Y., Cui, L., Huang, S., Wei, F., Li, Z. and Zhou, M. 2020 DocBank: A benchmark dataset for document layout analysis. Proceedings of the 28th International Conference on Computational Linguistics (COLING). 949-960.

[22] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P. and Bengio, Y. 2018 Graph attention networks. Proceedings of the 6th International Conference on Learning Representations (ICLR). Vancouver, Canada.

[23] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J. and Houlsby, N. 2021 An image is worth 16x16 words: Transformers for image recognition at scale. Proceedings of the 9th International Conference on Learning Representations (ICLR).

[24] Lin, T.-Y., Dollár, P., Girshick, R., He, K., Hariharan, B. and Belongie, S. 2017 Feature pyramid networks for object detection. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR). 2117-2125.

[25] Hendrycks, D. and Gimpel, K. 2016 Gaussian error linear units (GELUs). arXiv preprint arXiv:1606.08415.

[26] Smock, B., Pesala, R. and Abraham, R. 2022 PubTables-1M: Towards comprehensive table extraction from unstructured documents. Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR). 4634-4642.

[27] Yu, W., Lu, N., Qi, X., Gong, P. and Xiao, R. 2020 PICK: Processing key information extraction from documents using improved graph learning-convolutional networks. Proceedings of the 25th International Conference on Pattern Recognition (ICPR). 4363-4370.

[28] Dobrica, L. and Niemelä, E. 2002 A survey on software architecture analysis methods. IEEE Transactions on Software Engineering. 28(7): 638-653.

[29] Kazman, R., Bass, L., Abowd, G. and Webb, M. 1994 SAAM: A method for analyzing the properties of software architectures. Proceedings of the 16th International Conference on Software Engineering (ICSE). 81-90.

[30] Kazman, R., Klein, M. and Clements, P. 2000 ATAM: Method for architecture evaluation. Technical Report CMU/SEI-2000-TR-004. Software Engineering Institute,

Carnegie Mellon University, Pittsburgh.

[31] Bass, L., Clements, P. and Kazman, R. 2012 Software Architecture in Practice, 3rd ed. Addison-Wesley, Boston.

[32] European Union 2024 Regulation (EU) 2024/1689 laying down harmonised rules on artificial intelligence (Artificial Intelligence Act). Official Journal of the European Union. L series, 12 July 2024.

[33] European Union 2016 Regulation (EU) 2016/679 (General Data Protection Regulation). Official Journal of the European Union. L 119: 1-88.

[34] Zhong, X., Tang, J. and Jimeno Yepes, A. 2019 PubLayNet: Largest dataset ever for document layout analysis. Proceedings of the International Conference on Document Analysis and Recognition (ICDAR). 1015-1022.

[35] Harley, A.W., Ufkes, A. and Derpanis, K.G. 2015 Evaluation of deep convolutional nets for document image classification and retrieval. Proceedings of the 13th International Conference on Document Analysis and Recognition (ICDAR). 991-995.

[36] Jaume, G., Ekenel, H.K. and Thiran, J.-P. 2019 FUNSD: A dataset for form understanding in noisy scanned documents. Proceedings of the ICDAR Workshop on Open Services and Tools for Document Analysis (ICDAR-OST). 1-6.

[37] Park, S., Shin, S., Lee, B., Lee, J., Surh, J., Seo, M. and Lee, H. 2019 CORD: A consolidated receipt dataset for post-OCR parsing. Document Intelligence Workshop at NeurIPS 2019. Vancouver, Canada.

[38] Mathew, M., Karatzas, D. and Jawahar, C.V. 2021 DocVQA: A dataset for VQA on document images. Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision (WACV). 2200-2209.

[39] Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G. et al. 2019 PyTorch: An imperative style, high-performance deep learning library. Advances in Neural Information Processing Systems (NeurIPS). 32: 8024-8035.

[40] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A. et al. 2020 Transformers: State-of-the-art natural language processing. Proceedings of EMNLP 2020: System Demonstrations. 38-45.

[41] Fey, M. and Lenssen, J.E. 2019 Fast graph representation learning with PyTorch Geometric. ICLR Workshop on Representation Learning on Graphs and Manifolds. New Orleans, USA.