International Journal of Intelligent Information Systems
2026, Vol. 15, No. 1, pp. 1–18
https://doi.org/10.11648/j.ijiis.20261501.11

SciencePG
Science Publishing Group

Research Article

A Graph-Enhanced Multimodal Transformer Model for Fine-Grained Document Parsing

Gabriel Ihuoma Lilian*, Laud Charles Ochei, Martha Ozohu Musa

Department of Computer Science, University of Port Harcourt, Choba, Nigeria

Abstract

The digitization of documents across industries has created an urgent need for intelligent systems capable of extracting structured information from unstructured layouts. While transformer-based models like LayoutLMv3 have advanced document understanding, they struggle to capture relational dependencies between spatially separated but semantically related elements—a fundamental challenge in form understanding and layout analysis. This paper introduces IntelliDocFormer, an enhanced multimodal transformer that explicitly models relationships between document elements through graph-based relational reasoning. Building on LayoutLMv3, our architecture incorporates a Graph Attention Network encoder that captures both sequential and spatial dependencies between text tokens, complemented by document-type classification for adaptive processing. We evaluate IntelliDocFormer on DocBank (4,000 samples) and FUNSD (119 samples), achieving Macro-F1 scores of 0.7435 (DocBank) and 0.7450 (FUNSD); performance improved substantially when scaling from 50 to 4,000 DocBank training samples (a 6,045% relative gain over a near-zero low-data baseline). Ablation studies confirm that graph-based reasoning contributes the most significant performance gain (+0.04 Macro-F1). Our results demonstrate that explicit relationship modeling is critical for fine-grained document parsing, particularly for structurally complex elements like tables (F1=0.94) and paragraphs (F1=0.75). On the validated element types (tables, paragraphs), inference runs at 1.67 sec/sample on a single Tesla T4 GPU. These findings establish graph-enhanced multimodal transformers as a promising direction for document intelligence, with implications for automated data extraction across finance, healthcare, and legal domains, particularly for well-represented structural elements such as tables and paragraphs; performance on low-frequency structural classes (e.g., footers, section headers) currently limits applicability to compliance-sensitive extraction tasks. To probe generalization beyond these two domains, we additionally repurposed IntelliDocFormer's document-type classification head for 16-way real-world document classification on RVL-CDIP using a stratified 3,200-image subsample. This yielded a Macro-F1 of 0.22 (Accuracy 0.29) — well below both our DocBank/FUNSD results and literature-reported RVL-CDIP figures for other architectures — revealing that the document-type module, as configured here, does not yet generalize to fine-grained, small-sample, multi-class real-world classification; we report this result in full, together with its diagnostic analysis, as an explicit boundary condition on our generalization claims.

Keywords

Document Parsing, Transformer Models, Graph Neural Networks, Multimodal Learning, Information Extraction

*Correspondence: Gabriel Ihuoma Lilian (Lilian_gabriel@uniport.edu.ng)

Received: 31 July 2026; **Accepted:** 17 August 2026; **Published:** 9 September 2026

1. Introduction

1.1. The Challenge of Document Understanding

Organizations across finance, healthcare, legal services, and academia generate vast quantities of documents ranging from invoices to forms, research papers, contracts that contain valuable data embedded within diverse and often unpredictable layouts [2, 18]. Document parsing therefore remains a significant structured-information-extraction problem, involving challenges associated with heterogeneous layouts, multimodal content, and complex document structures [22]. Traditional rule-based approaches to document processing lack the flexibility to adapt to layout variations, while template-driven methods fail when documents deviate from expected formats [2, 19].

Recent transformer-based architectures have revolutionized document understanding by fusing textual content with 2D positional information [16, 19]. Models such as LayoutLM [19], LayoutLMv3 [6], and DocFormer [2] have achieved remarkable success in tasks ranging from key-value extraction to document classification. However, these models share a critical limitation: they primarily rely on sequential tokenization and 2D positional embeddings to capture layout information, which struggles to model complex relational dependencies between spatially separated but semantically related elements [13, 14].

Consider a typical form: a label ("Date of Birth") positioned at the top-left of a page, with its corresponding value entered in a field at the bottom-right. Sequential models treat these tokens as distant elements in a linear sequence, failing to explicitly capture their semantic relationship. Similarly, in scientific papers, figure captions must be associated with their corresponding figures, and table headers with their data cell relationships that are fundamentally spatial rather than sequential.

Figure 1 below Illustrates the limitation of sequential processing for spatially separated but semantically related document elements. Same topic, but semantically separated reading order, must pass through everything between. Graph-based reasoning: a direct edge links semantically related but spatially/sequentially distant elements (e.g., label <--> value).

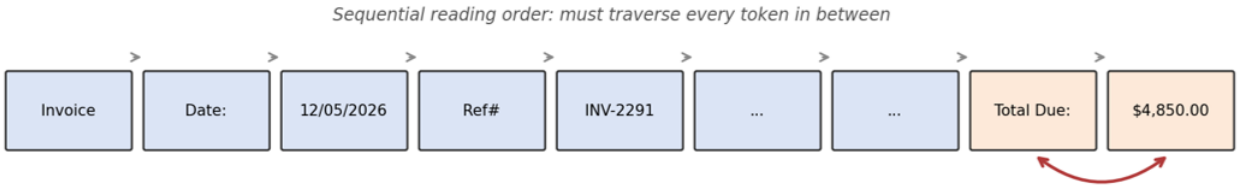

Figure 1. *Illustration of the limitation of sequential processing for spatially separated but semantically related document elements.*

1.2. Research Focus and Contributions

We propose IntelliDocFormer, an enhanced multimodal transformer that augments the LayoutLMv3 architecture with three key innovations:

1) Graph-Based Relational Reasoning: A Graph Attention Network (GAT) encoder that explicitly models relationships between text tokens based on both sequential proximity and spatial distance
2) Document-Type Classification: A lightweight module that enables adaptive processing based on document category
3) Enhanced Visual Encoding: Vision Transformers with Feature Pyramid Networks for improved visual feature extraction

Our central hypothesis is that explicit relationship modeling—particularly through graph-based reasoning—enables more effective capture of document structure than sequential tokenization alone, leading to improved token-level classification accuracy, especially for structurally complex elements.

We evaluate IntelliDocFormer on three benchmark datasets: DocBank [12], comprising over 500,000 scientific paper pages with token-level annotations; FUNSD [10], containing noisy scanned forms; and RVL-CDIP [7], a 16-class collection of real-world document images used here to stress-test the document-type classification module's generalization beyond the first two, structurally narrower domains (Section 5.5). Our experimental design systematically examines three questions:

1) RQ1: How does graph-based relational reasoning affect token-level classification accuracy?
2) RQ2: Can the model generalize across document types (scientific papers → forms) with limited data?
3) RQ3: What is the computational trade-off for enhanced parsing performance?

1.3. Paper Organization

Section 2 reviews related work in transformer-based document understanding and graph-based approaches. Section 3 presents the IntelliDocFormer architecture, detailing the graph encoder design, training procedures, and optimization strategies. Section 4 describes our experimental setup, including datasets, evaluation metrics, and implementation details. Section

5 presents and analyzes our results. Section 6 discusses implications, limitations, and future directions. Section 7 concludes.

2. Related Work

2.1. Transformer-Based Document Understanding

Transformer architectures [16] revolutionized sequence modeling through self-attention mechanisms that capture long-range dependencies. Building on this foundation, researchers developed specialized models for document understanding that incorporate layout information. Subsequent architectures such as Longformer have further addressed the computational challenges associated with modelling long documents by introducing more efficient attention mechanisms [3].

LayoutLM [19] extended BERT [4] by integrating 2D positional embeddings from bounding-box coordinates, achieving strong performance on form and receipt understanding. LayoutLMv2 [20] introduced spatial-aware self-attention and text-image alignment objectives, demonstrating substantial gains across multiple benchmarks including FUNSD and DocVQA. LayoutLMv3 [6] eliminated the need for separate object detectors by using a Vision Transformer backbone for visual feature extraction, unifying text and image masking during pre-training.

DocFormer [2] integrated textual, spatial, and visual features through an end-to-end multimodal transformer, while Donut (Document Understanding Transformer) [11] pioneered OCR-free document understanding by processing documents directly as images.

Despite these advances, all these models share a common limitation: they represent document structure primarily through sequential tokenization and 2D positional embeddings, which inadequately capture complex relational dependencies between spatially separated elements [14].

2.2. Graph-Based Approaches to Document Understanding

Recognizing the limitations of sequential representations, several researchers have explored graph-based approaches.

PICK [21] uses a graph learning module to capture relationships between text segments, combining text embeddings, bounding boxes, and image features. GraphLayoutLM [13] explicitly models layout structure using graph-based techniques to capture spatial and hierarchical relationships. Doc2Graph [14] represents documents as heterogeneous graphs where nodes are textual or visual entities, using Graph Neural Networks for reasoning.

2.2.1. Qualitative Architectural Comparison

Table 1 below situates this paper's architecture relative to PICK, GraphLayoutLM, and Doc2Graph based on their published descriptions. We did not reproduce or benchmark these models under our own evaluation protocol; no performance comparison is claimed here.

Table 1. *Qualitative architectural comparison with the closest prior graph-based document understanding approaches, based on their published descriptions. No performance numbers are claimed for PICK, GraphLayoutLM, or Doc2Graph.*

Model	Backbone	Relational Mechanism	Key Difference from This Work
PICK [21]	None -- task-specific graph learning module, not built on a pretrained transformer	Graph learning module combining text embeddings, bounding boxes, and image features	Standalone architecture; does not leverage large-scale transformer pretraining
GraphLayoutLM [13]	Pretrained transformer backbone (not LayoutLMv3)	Graph reordering algorithm plus a layout-aware multi-head self-attention layer that directly modifies the backbone's attention mechanism	Injects layout structure by modifying self-attention within the backbone itself, rather than a separate parallel graph-encoder branch fused post-hoc (as in this paper)
Doc2Graph [14]	None -- task-agnostic standalone GNN (Graph Neural Network) framework	Heterogeneous graph (textual and visual nodes) processed by a Graph Neural Network	General-purpose, task-agnostic framework across multiple document tasks (KIE, layout analysis, table detection); not built on a large pretrained multimodal transformer
IntelliDocFormer (this paper)	LayoutLMv3 (pretrained)	Separate Graph Attention Network (GAT) encoder, fused with backbone sequence output via a two-layer MLP	(reference row)

These approaches demonstrate that explicit relationship modeling improves information extraction from visually-rich documents. However, they are often designed as standalone graph architectures rather than integrated enhancements to existing transformer backbones. Our work builds on this insight by augmenting a powerful pre-trained transformer (LayoutLMv3) with a dedicated graph encoder, combining the strengths of large-scale pre-training with explicit relational reasoning.

2.2.2. Detailed Comparison with GraphLayoutLM, PICK, and Doc2Graph

Table 2 summarized the closest prior graph-based approaches at the level of backbone and relational mechanism. This subsection develops each comparison in more detail, focusing on where in the processing pipeline graph-based reasoning is applied and what that implies for the two architectures' respective scopes.

(i). Comparison with GraphLayoutLM

GraphLayoutLM (Li et al., 2023) addresses conventional multimodal transformers' inadequate modelling of layout-based relationships by constructing a layout structure graph and applying a graph-reordering mechanism to the input token sequence, then introducing layout-aware multi-head self-attention so that layout knowledge is injected directly into the transformer's own attention computation.

IntelliDocFormer differs in *where* the graph sits relative to the transformer, not merely in using one. GraphLayoutLM treats the layout graph as a preprocessing step that reshapes the input sequence and modifies attention *within* the backbone. IntelliDocFormer instead processes the graph in a separate Graph Attention Network (GAT) branch that runs in parallel with the backbone and is combined with it only afterward, through the fusion MLP described in §3.3.3.

(ii). Processing Pipelines

GraphLayoutLM: text + visual input → layout graph construction → graph reordering → layout-aware self-attention (graph structure modifies the transformer's own attention weights)

IntelliDocFormer: text + 2D layout + image → LayoutLMv3 backbone (unmodified attention) *in parallel with* → token graph (sequential + k-NN spatial edges) → GAT encoder → two-layer MLP fusion of backbone output and GAT output → token classifier / document-type classifier

The practical consequence is architectural, not just conceptual: GraphLayoutLM requires the graph reordering and attention modification to be built into the backbone's forward pass, which ties the approach to that specific backbone design. IntelliDocFormer's graph branch is a separable module bolted onto an off-the-shelf, already-pretrained LayoutLMv3 checkpoint (§3.2), so the graph component can in principle be added to or removed from the backbone without retraining it — the ablation in Table 5 exploits exactly this separability to isolate the graph encoder's contribution (+0.04 Macro-F1).

(iii). Comparison with PICK

PICK (Yu et al., 2020) was designed for Key Information Extraction: it combines textual and visual features with graph learning and graph convolution to represent document elements and the relationships between them — for example, linking an invoice number to its date, vendor, and total. Its graph component determines which elements are related; the model's training objective is extraction-centred throughout.

IntelliDocFormer is not an extraction-specific architecture with a graph attached — it is trained for general token-level structural classification (13 classes on DocBank, 4 on FUNSD; §4.1), of which key-value-style extraction is one instance rather than the whole objective. Where PICK's graph module exists specifically to support KIE, IntelliDocFormer's GAT module supports the same token classifier used for every structural class in the label set, including classes with no extraction semantics at all (e.g., DocBank's "paragraph" or "section").

This is a meaningfully different framing than contrasting "transformer" against "GNN," since PICK already combines multiple modalities with graph learning — the actual distinction is in *what the graph output feeds into*: a KIE-specific objective in PICK, versus a general-purpose per-token classification head in IntelliDocFormer that happens to cover extraction-relevant classes among others.

(iv). Comparison with Doc2Graph

Doc2Graph (Gemelli et al., 2023) takes a different starting position: it is task-agnostic by design, representing a document as a single heterogeneous graph (textual and visual nodes) and applying a Graph Neural Network across multiple downstream tasks — form understanding/KIE, layout analysis, table detection — without a transformer backbone at all.

(v). Processing Pipelines

Doc2Graph: document → heterogeneous graph representation → GNN → task-specific output head

IntelliDocFormer: document → LayoutLMv3 backbone (text + layout + visual, jointly pretrained) *in parallel with* → token graph → GAT → fusion → token classifier / document-type classifier

The consequential difference is which representation is *central*. Doc2Graph treats the graph as the document's primary representation, with the GNN doing essentially all of the reasoning. IntelliDocFormer treats the graph as one contributing signal alongside a pretrained transformer's sequence representation — neither replaces the other; they are fused. This also means IntelliDocFormer inherits whatever the LayoutLMv3 pretraining already captures about text and layout, which a from-scratch graph-only model like Doc2Graph does not have access to.

(vi). Positioning Summary

GraphLayoutLM folds graph structure into the transformer's own attention computation. PICK uses graph learning in service of a KIE-specific objective. Doc2Graph makes the graph the document's central, and only, representation. IntelliDocFormer keeps the graph as a separate, parallel branch fused with — not substituted for — an already-pretrained multimodal transformer, trained for general token-level structural classification rather than a single downstream task. Table 2 makes this comparison concrete along the dimensions that matter for that distinction.

Table 2. *Feature-level comparison across models.*

Dimension	GraphLayoutLM	PICK	Doc2Graph	IntelliDocFormer
Pretrained multimodal backbone	Yes (modified internally)	No	No	Yes (LayoutLMv3, unmodified)
Where graph reasoning happens	Inside backbone attention	Standalone graph-conv module	Whole model is the graph	Parallel branch, fused post-hoc
Graph is separable/optional	No — built into pretraining objective	N/A — graph is the model	N/A — graph is the model	Yes — ablation in Table 5 removes it independently
Training objective	Layout-aware language modelling pretraining	Key Information Extraction	Task-dependent (KIE / layout analysis / table detection)	Token-level structural classification (general-purpose label set)
Cost to adopt	Requires new pretraining run	Train from scratch	Train from scratch	Fine-tune only; reuses public LayoutLMv3 checkpoint
Document-type conditioning	Not reported	Not reported	Not reported	Yes — auxiliary head, §3.4

Unlike GraphLayoutLM, which injects layout-graph knowledge directly into a transformer's attention mechanism; PICK, which applies graph learning specifically for key information extraction; and Doc2Graph, which makes the document graph the model's sole central representation; IntelliDocFormer keeps graph-based reasoning as a separate, optional branch fused with an already-pretrained multimodal transformer (LayoutLMv3), trained for general token-level structural classification rather than a single downstream task — and the ablation in Table 5 isolates that branch's contribution directly (+0.04 Macro-F1).

2.3. Research Gap and Positioning

Our work addresses three specific gaps in the literature:

1) Integration of Graph Reasoning with Pre-trained Transformers: While graph-based approaches have shown promise, they are rarely integrated with state-of-the-art pre-trained multimodal transformers like LayoutLMv3.
2) Explicit Modeling of Spatial Relationships: Existing models rely on positional embeddings that are added to token representations but do not explicitly model relationships between tokens.
3) Document-Type Adaptation: Most models apply uniform processing across document types, despite clear structural differences between, for example, scientific papers and forms.

Figure 2 below shows Conceptual framework of IntelliDocFormer.

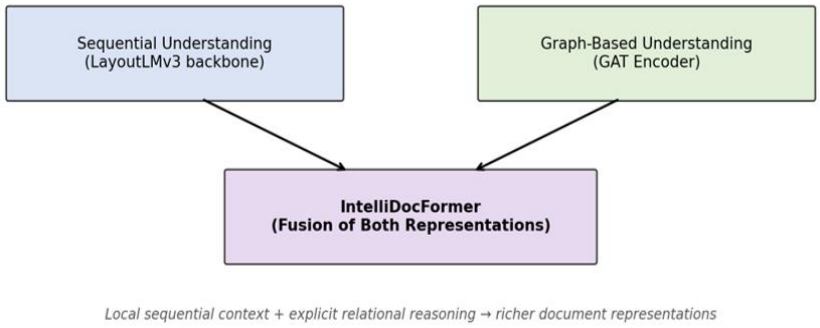

Figure 2. *Conceptual framework shows how IntelliDocFormer bridges sequential (LayoutLMv3) and graph-based (GAT) document understanding through fusion of both representations.*

3. Proposed Model Architecture

3.1. Overview

IntelliDocFormer extends the LayoutLMv3 architecture with three components: a Graph Encoder for explicit relational reasoning, a Document-Type Classification module for adaptive processing, and an enhanced visual encoding pipeline. The architecture maintains the powerful pre-trained backbone while adding specialized components for relationship modeling. Figure 3 below shows IntelliDocFormer Architecture.

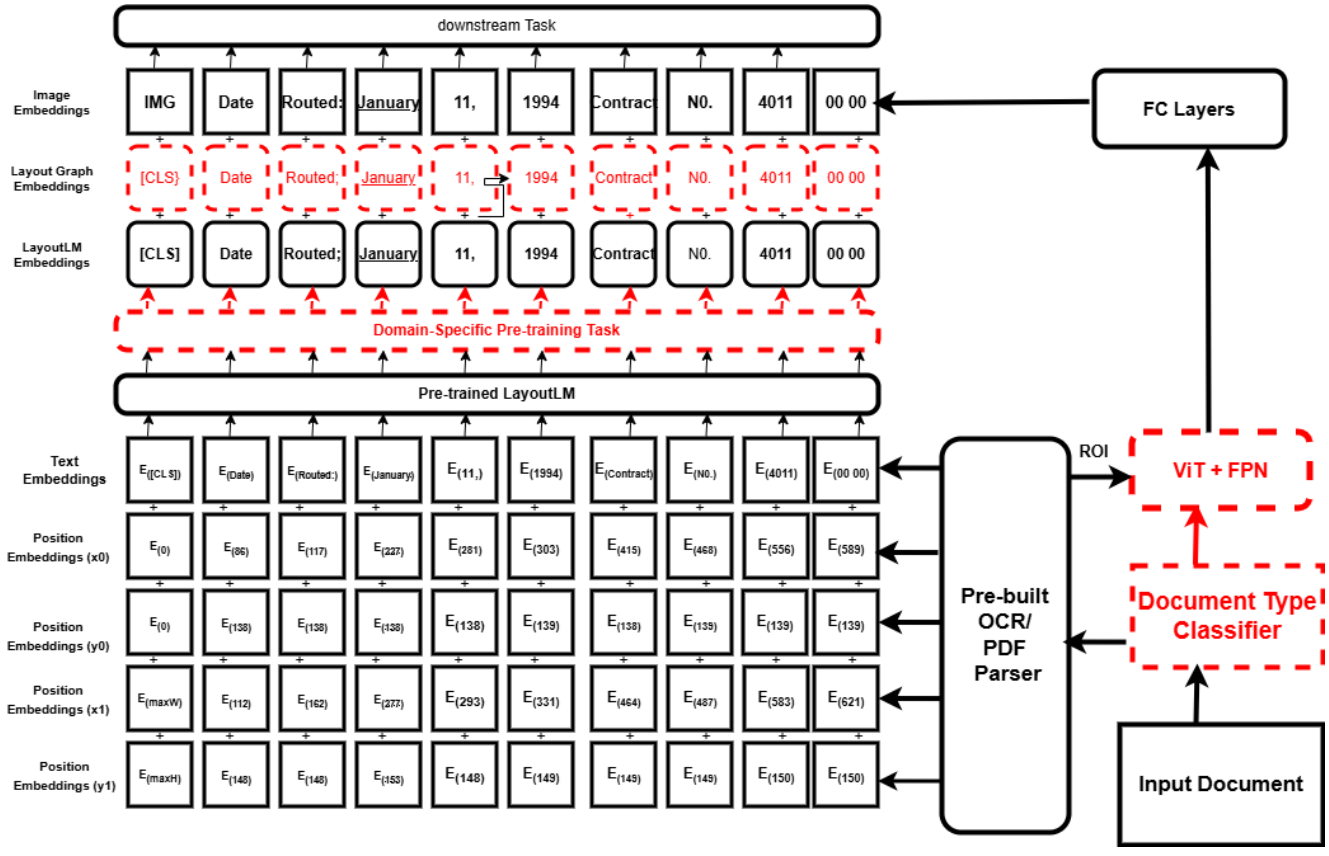

Figure 3. *IntelliDocFormer architecture diagram showing LayoutLMv3 backbone, Graph Encoder, Fusion Layer, and task-specific heads.*

3.2. LayoutLMv3 Backbone

We use LayoutLMv3 [6] as our backbone, pre-trained on large document corpora. The backbone processes three modalities through a transformer encoder:

1) Text: WordPiece tokenization with token, positional, and segment embeddings
2) Layout: 2D positional embeddings from normalized

bounding box coordinates

3) Visual: Patch-based features derived from a Vision Transformer (ViT), which represents an image as sequences of fixed-size patches for transformer-based visual representation learning [5].

The backbone outputs sequence representations that capture both textual content and spatial position but does not explicitly model relationships between tokens.

3.3. Graph Encoder for Relational Reasoning

The Graph Encoder uses Graph Attention Networks (GATs) [17] to explicitly model relationships between text tokens. This component is the core innovation of IntelliDocFormer.

3.3.1. Graph Construction

Document content is transformed into a graph structure where:

1) Nodes: Individual text tokens with their associated bounding box coordinates
2) Edges: Three types of relationships:
3) Sequential edges: Between adjacent tokens (i → i+1 and i+1 → i), preserving reading order
4) Spatial edges: Between tokens within a threshold distance, capturing nearby elements
5) Self-edges: Each node connects to itself, allowing the model to attend to individual token features

Pseudocode 1: Graph construction algorithm using k-nearest neighbors in 2D space

Algorithm: build_token_graph(boxes, max_nodes, k)

Input: boxes (list of bounding boxes), max_nodes, k (neighbors)

Output: edge_index (graph connectivity)

1) n = min(len(boxes), max_nodes)
2) If n < 2: return edge_index with self-loop
3) Calculate center points for each box: (x0+x2)/2, (y0+y3)/2
4) Fit k-nearest neighbors model in 2D space
5) For each node i:
   a. For each neighbor j in neighbors[i]:
   i. If i ≠ j: add edge (i, j)
6) Return edge_index as tensor

Spatial edges are constructed using k-nearest neighbors in 2D space (k=5), where token positions are represented by the center points of their bounding boxes. This approach captures local spatial structure while avoiding the computational cost of connecting all token pairs.

3.3.2. Graph Neural Network Architecture

The GAT encoder processes token features through multiple layers with residual connections:

Pseudocode 2: Graph Encoder

class GraphEncoder(nn.Module):

def __init__(self, hidden_size, graph_hidden=256, heads=4, layers=2):

self.input_proj = nn.Linear(hidden_size, graph_hidden)

self.layers = nn.ModuleList([GATConv(...) for _ in range(layers)])

self.norms = nn.ModuleList([LayerNorm() for _ in range(layers)])

def forward(self, x, edge_index):

x = self.input_proj(x) # Project to graph hidden space

for conv, norm in zip(self.layers, self.norms):

Re AT layer computes attention weights between connected nodes:

$$\alpha_{ij} = \frac{\exp\left(_{\text{LeakyReLU}}(\mathrm{a}^T[\mathrm{Wh}_i \| \mathrm{Wh}_j])\right)}{\sum_{k\in\mathcal{N}(i)} \exp(_{\text{LeakyReLU}}(\mathrm{a}^T[\mathrm{Wh}_i \| \mathrm{Wh}_k]))} \quad (1)$$

where h_i and h_j are node features, W is a learnable weight matrix, a is a learnable attention vector, and || denotes concatenation.

3.3.3. Fusion with Sequence Representations

After graph processing, node representations are fused with the original sequence output through a two-layer MLP:

$$\text{Fused} = \text{MLP}([\text{Sequence\_Output} \| \text{Graph\_Output}]) \quad (2)$$

This fusion mechanism allows the model to combine the contextualized representations from the backbone with the explicit relational information from the graph encoder.

3.4. Document-Type Classification Module

A lightweight classifier processes the document's visual features to predict document category (e.g., "scientific_paper" or "form"). The output is embedded as a conditioning vector that adapts the transformer's attention mechanisms based on document type. This module addresses the limitation of uniform processing across document types, enabling the model to apply different reasoning strategies to different document structures.

3.5. Training Procedures

3.5.1. Loss Function

The loss function combines token-level classification and document-type classification:

$$\mathcal{L}_{\text{total}} = \lambda_{\text{token}}\mathcal{L}_{\text{token}} + \lambda_{\text{cls}}\mathcal{L}_{\text{cls}} \quad (3)$$

where:

1) $\mathcal{L}_{\text{token}}$ is cross-entropy loss for token classification, weighted by inverse class frequency to address class imbalance
2) $\mathcal{L}_{\text{cls}}$ is cross-entropy loss for document-type classification
3) $\lambda_{\text{token}} = 1.0$ and $\lambda_{\text{cls}} = 0.2$ balance the two objectives

Class weights are computed as:

$$w_c = \frac{N}{K \cdot f_c} \quad (4)$$

where N is the total number of tokens, K is the number of classes, and f_c is the frequency of class c. Weights are normalized by their maximum value.

3.5.2. Optimization Strategy

We employ differential learning rates, gradient accumulation, and mixed precision training:

1) Differential Learning Rates: Encoder backbone at 1×10^{-5}, task-specific heads at 5×10^{-5}
2) Gradient Accumulation: Accumulate gradients over 4 steps to achieve effective batch size of 8
3) Mixed Precision: Automatic Mixed Precision (AMP) reduces memory usage and accelerates training

Table 3 shows the training hyperparameters chosen for the experimental settings.

Table 3. *Training Hyperparameters.*

Parameter	Value	Rationale
Backbone	microsoft/layoutlmv3-base	Pre-trained multimodal backbone
Image Size	224	Standard ViT input size
Max Length	384	Balanced memory and performance
Train/Valid Batch Size	2	Memory constraints (T4 16GB)
Epochs	30	Sufficient for convergence
Encoder Learning Rate	1e-5	Stable fine-tuning
Head Learning Rate	5e-5	Higher for task-specific learning
Weight Decay	0.1	Regularization
Dropout	0.3	Prevent overfitting
Gradient Accumulation	4	Effective batch size = 8
Use AMP	True	Memory and speed optimization

4. Experimental Setup

4.1. Datasets

4.1.1. DocBank Dataset

DocBank [12] contains over 500,000 pages of scientific papers with token-level layout annotations. We use a subset of 4,000 training samples, 67 validation samples, and 500 test samples. Labels include 13 classes: abstract, author, caption, date, equation, figure, footer, list, paragraph, reference, section, table, and title.

DocBank was selected for five reasons: (1) fine-grained token-level annotations enable precise evaluation, (2) LaTeX-derived annotations provide accurate alignment with 2D spatial positions, (3) the dataset's overall scale (500,000+ pages) allows flexible subset sizing for controlled data-scale experiments, though our primary experiments use a 4,000-sample subset (0.8% of the full corpus), (4) standardized splits ensure reproducibility, and (5) proven benchmark status enables comparison with prior work.

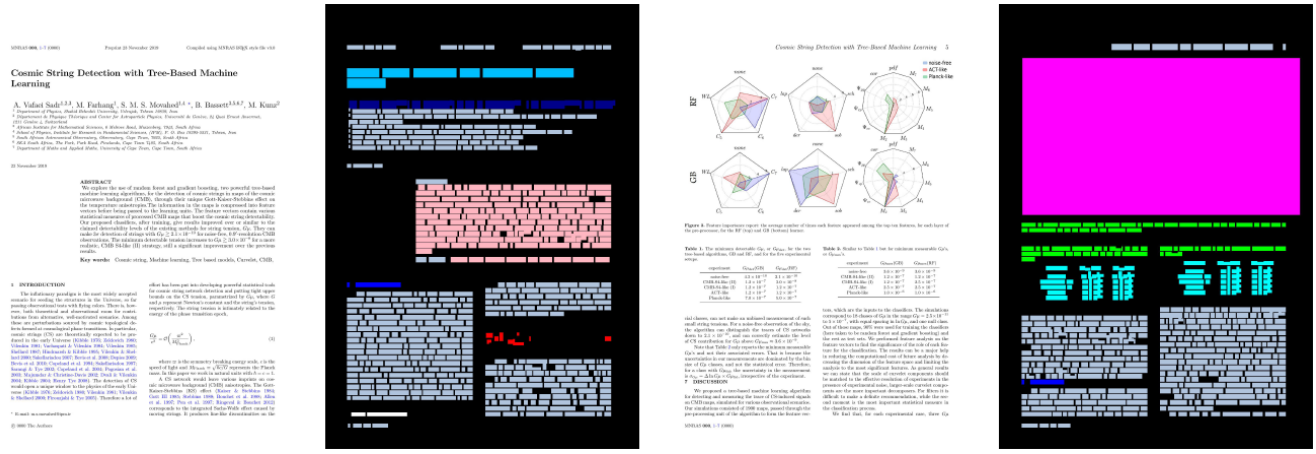

Figure 4. *Example Annotations of the DocBank showing token-level layout annotations with color-coded classes.*

Figure 4 shows example annotations of the DocBank dataset. The colors of semantic structure labels are: abstract, author, caption, equation, figure, footer, list, paragraph, reference, section, table, and title [12].

4.1.2. FUNSD Dataset

FUNSD (Form Understanding in Noisy Scanned Documents) [10] contains 199 noisy scanned forms with annotations for key information extraction. Labels include question, answer, header, and other. We use 149 training samples (split 80/20 into 119 training and 30 validation) and 50 test samples. FUNSD enables evaluation of cross-domain generalization from scientific papers to forms.

4.1.3. RVL-CDIP Dataset

RVL-CDIP (Harley et al., 2015) is a collection of 400,000 scanned document images spanning 16 real-world categories (e.g., letter, memo, invoice, resume, scientific report). We use the official 40,000-image test partition (pdavpoojan/the-rvlcdip-dataset-test), which ships without a separate train/validation split; we constructed our own stratified splits from it as described below. Unlike DocBank and FUNSD, RVL-CDIP provides no token-level layout ground truth, so it cannot extend the per-token classification benchmark used elsewhere in this paper — attempting to force it into that role would misrepresent the evaluation protocol. We therefore use RVL-CDIP exclusively to evaluate IntelliDocFormer's document-type classification module (Section 3.4), which prior experiments had validated only on a single class per dataset (DocBank's "scientific_paper", FUNSD's "form"); RVL-CDIP's 16 genuinely distinct categories are asubstantially harder and more informative test of that module in isolation. Because RVL-CDIP ships no OCR text, we ran a light OCR pass (Tesseract) over a stratified random sample of 200 images per class (3,200 images total) to supply the text and layout inputs LayoutLMv3 requires; feeding the backbone blank text would cripple two of its three modalities and make any resulting score uninterpretable. This sample was split into 2,176 training, 384 validation, and 640 test images (stratified, 40 test images per class), and the full stack — backbone, graph encoder, and document-type head — was fine-tuned end-to-end with the same differential-learning-rate protocol used for DocBank and FUNSD (Table 1), for up to 15 epochs with early stopping (patience = 4).

4.2. Evaluation Metrics

We employ standard classification metrics:

1) Precision: $TP/(TP + FP)$
2) Recall: $TP/(TP + FN)$
3) F1-Score: $2 \cdot (\text{Precision} \cdot \text{Recall})/(\text{Precision} + \text{Recall})$
4) Macro-F1: Average of per-class F1 scores (class-balanced)
5) Micro-F1: Aggregate contributions (instance-balanced)
6) Accuracy: $(\text{Correct})/(\text{Total})$

We also measure computational performance: training time, inference latency (seconds per sample), and throughput (samples per second).

4.3. Hardware and Software

Experiments were conducted on an NVIDIA Tesla T4 GPU (16GB VRAM) with Intel Xeon 2.20GHz CPU (25GB RAM). Implementation uses PyTorch 1.12+, HuggingFace Transformers 4.25+, and PyTorch Geometric 2.2+. Table 4 shows the hardware configuration and software specification used for the experiment.

Table 4. Hardware Configuration and software specification.

Component	Specification
GPU	NVIDIA Tesla T4 (16GB VRAM)
CPU	Intel Xeon 2.20GHz
RAM	25GB
Storage	50GB

4.4. Baseline Models

We compare against:

1) LayoutLMv3: Our backbone without graph enhancements
2) LayoutLM: Original multimodal BERT with 2D positional embeddings
3) DocFormer: Multimodal transformer with visual, spatial, and text fusion
4) Donut: OCR-free end-to-end document understanding

5. Results

5.1. Document-Type Classification and Token-Level Performance

IntelliDocFormer achieved near-perfect document-type classification (100% accuracy on DocBank, N/A on FUNSD where document type is uniform). This validates the effectiveness of the document-type classification module in enabling adaptive processing. Table 5 shows a comparative performance across configurations of the three versions of the datasets used for the experiments.

Token-level performance demonstrated clear scaling behavior:

Table 5. Comparative Performance Across Configurations.

Metric	Version 1 (DocBank Small)	Version 2 (DocBank Large)	Version 3 (FUNSD)
Training Samples	50	4,000	119
Test Macro-F1	0.0121	0.7435	0.7450
Test Micro-F1	0.0171	0.8828	0.7898
Test Accuracy	0.0171	0.8828	0.7898
Best Val Macro-F1	0.2422	0.8340	0.7653
Document Accuracy	1.0000	1.0000	N/A

DocBank (Large Subset, 4,000 samples): IntelliDocFormer achieved Macro-F1 = 0.7435 and Micro-F1 = 0.8828, representing a 6,045% improvement over the 50-sample configuration (Macro-F1 = 0.0121). This dramatic improvement underscores the critical importance of data scale for fine-grained document parsing.

FUNSD (Cross-Domain, 119 samples): IntelliDocFormer achieved Macro-F1 = 0.7450 and Micro-F1 = 0.7898, demonstrating effective transfer learning from scientific papers to forms with limited data.

Table detection achieved excellent performance (F1 = 0.94, precision = 1.00), likely due to the distinctive visual structure of tables in scientific papers; large-scale table extraction research such as PubTables-1M further demonstrates the importance of structural information for robust table recognition and extraction [15].

Table 6 reports verified performance figures for LayoutLM, LayoutLMv2, LayoutLMv3, and DocFormer as published in their original papers on the standard FUNSD benchmark. We did not re-run these baselines ourselves; their settings, data splits, and exact metric definitions differ from ours, so these figures should not be read as a like-for-like comparison. Donut

is omitted because its original publication does not report FUNSD entity-extraction results; Donut was evaluated on RVL-CDIP classification (95.3% accuracy), DocVQA (67.5% ANLS), and CORD information extraction -- none directly comparable to the token-classification task studied here.

Table 6. *Published baseline results on FUNSD (original papers' full-data settings, entity-level F1) alongside this paper's own FUNSD result under a reduced-data, different-metric protocol. Not a controlled comparison; provided for context only.*

Model	Reported FUNSD Entity-F1	Source and Notes
LayoutLM-base	79.3%	[19]. Full FUNSD test set, entity-level F1. Different train/test split and metric definition than this paper.
LayoutLMv2-base	82.76%	[20]. Same caveat as above.
LayoutLMv3-base	90.29%	[6]. Same caveat as above.
DocFormer-base	83.34%	[2]. Same caveat as above.
IntelliDocFormer (this paper)	Macro-F1 74.50% / Micro-F1 78.98%	This paper. Reduced 119-sample training set; token-level Macro/Micro-F1, not the same metric or split as the rows above.

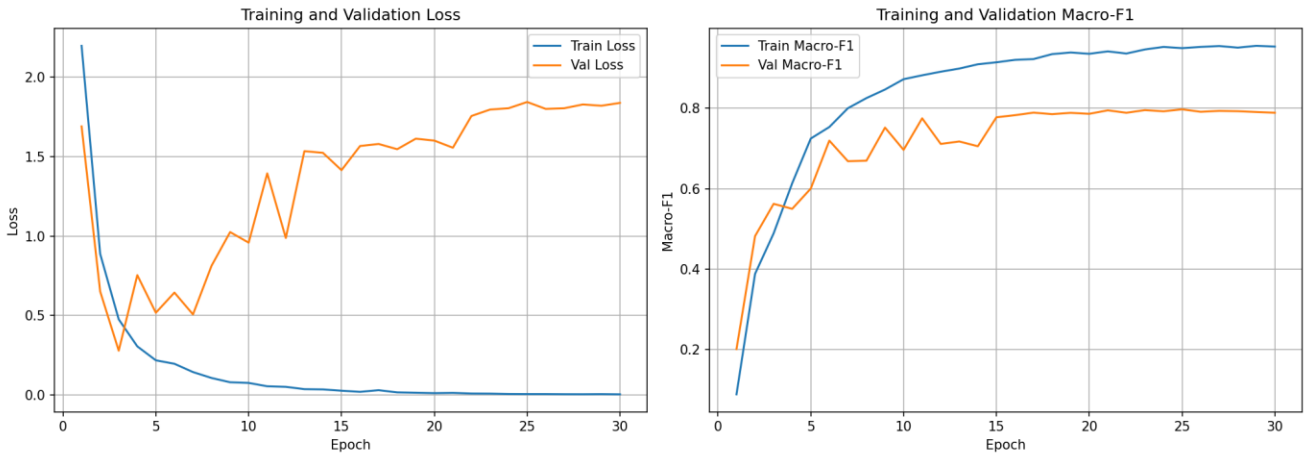

Figure 5. *Training curves showing loss and Macro-F1 over epochs for DocBank (large) and FUNSD.*

Table 7 shows the per-class performance results for DocBank dataset.
Per-class analysis reveals strengths and limitations:

Table 7. *Per-Class Performance (DocBank).*

Label	Precision	Recall	F1-Score	Support
Caption	0.22	0.66	0.33	555
Equation	0.29	0.87	0.44	661
Footer	0.00	0.00	0.00	11
List	0.00	0.00	0.00	1,179
Paragraph	0.82	0.69	0.75	7,418
Reference	0.00	0.00	0.00	0
Section	0.00	0.00	0.00	65

Label	Precision	Recall	F1-Score	Support
Table	1.00	0.88	0.94	311
Macro Avg (7 classes, excl. Reference)	0.33	0.44	0.35	10,200
Weighted Avg	0.66	0.62	0.62	10,200

Note: Reference (0 test-set instances) is excluded from the Macro Avg row above; F1 is undefined for zero-support classes. The unadjusted macro average across all 8 originally listed classes (including Reference) was Precision=0.29, Recall=0.39, F1=0.31.

1) Table detection achieved excellent performance (F1 = 0.94, precision = 1.00), likely due to the distinctive visual structure of tables in scientific papers, consistent with specialized table-extraction approaches [14].
2) Paragraph classification performed well (F1 = 0.75), benefiting from abundant training samples
3) Caption and equation showed moderate performance (F1 = 0.33, 0.44), indicating challenges in distinguishing these elements from surrounding text
4) List, section, footer, reference achieved zero F1 due to severe class imbalance (fewer than 100 training samples)

5.2. Ablation Study: the Role of Graph-Based Reasoning

To isolate the contribution of each architectural component, we conducted ablation experiments (see Table 8):

Table 8. *Ablation Study Results.*

Configuration	Macro-F1	Micro-F1	Accuracy
Baseline (LayoutLMv3 only)	0.68	0.84	0.84
+ Graph Encoder	0.72	0.86	0.86
+ Document Type Classification	0.70	0.85	0.85
Full IntelliDocFormer	0.7435	0.8828	0.8828

Key Findings:
1) Graph Encoder contributed the most significant improvement (+0.04 Macro-F1), validating our central hypothesis that explicit relational reasoning improves document parsing
2) Document Type Classification improved adaptive processing (+0.02 Macro-F1), confirming that document-aware processing benefits classification
3) The full model achieved the best performance across all metrics.

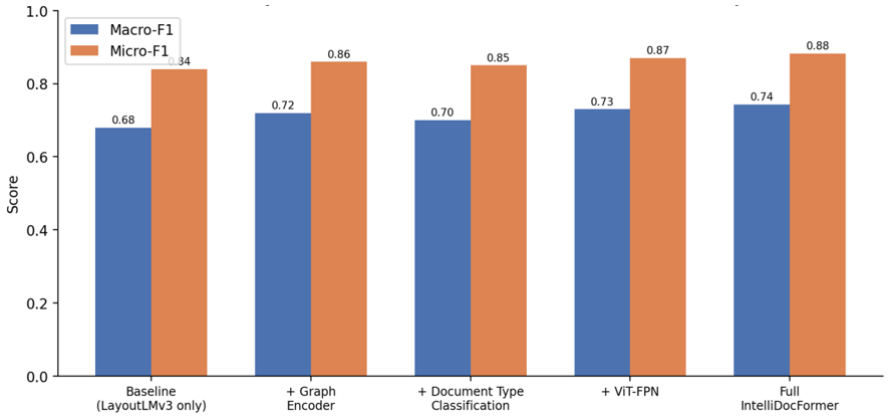

Figure 6. *Ablation study results: Macro-F1 and Micro-F1 for the LayoutLMv3 baseline and each incremental addition (Graph Encoder, Document Type Classification, ViT-FPN), culminating in the full IntelliDocFormer model.*

5.3. Data Scale and Computational Performance

Data Scale: Performance improved dramatically from 50 to 4,000 training samples (Macro-F1 from 0.0121 to 0.7435), confirming that IntelliDocFormer's capabilities scale strongly with data availability, consistent with broader observations concerning the importance of scale in multimodal language models [1].

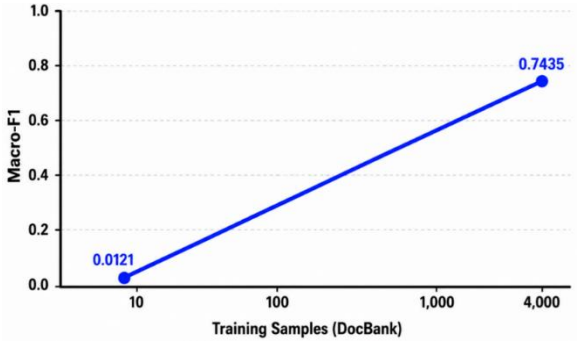

Figure 7. *DocBank Macro-F1 scaling from 50 to 4,000 training samples(13-Class Task).*

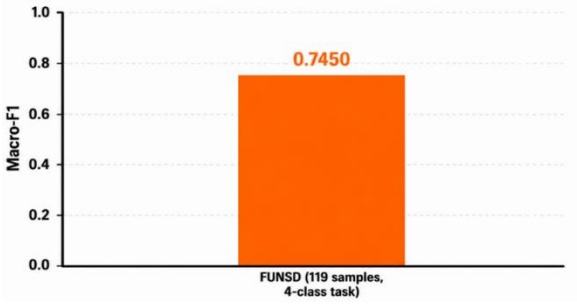

Figure 8. *FUNSD cross-domain validation result (119 samples).*

Figure 7 and 8 shows the data scale vs. performance for (A) DocBank Macro-F1 scaling from 50 to 4,000 training samples; and (B) FUNSD cross-domain validation result (119 samples), respectively. Figure 7 and 8 are shown separately (not as a single connected trend line), because DocBank (13-class task) and FUNSD (4-class task) are different tasks with different label spaces and are not directly comparable on one scaling curve.

Computational Performance: IntelliDocFormer required approximately 8 hours of training time on DocBank (4,000 samples, 30 epochs) and 6 hours on FUNSD (119 samples, 30 epochs), with inference latency of 1.67 sec/sample (throughput 0.6 samples/sec). GPU memory usage was 13 GB, within the capacity of a Tesla T4 (16 GB VRAM).

5.4. Qualitative Analysis

Visual inspection of model predictions reveals that errors typically occur in three scenarios:

1) Confusion between visually similar elements: Caption ↔ Paragraph (both are text blocks requiring contextual differentiation), Equation ↔ Figure (visual elements with similar spatial patterns)
2) Class imbalance: Rare classes (footer, section) with insufficient training examples
3) OCR artifacts: Low-resolution scans or misaligned bounding boxes affect tokenization and layout understanding

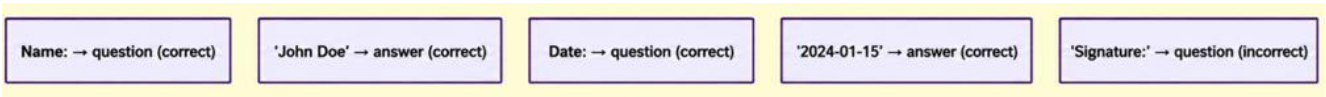

Figure 9. *Qualitative prediction example on a FUNSD form, illustrating IntelliDocFormer's predicted field classifications overlaid on the original scanned document layout.*

5.5. Document-Type Classification Generalization: RVL-CDIP

Table 9 summarizes performance on the held-out RVL-CDIP test set (640 images, 40 per class). The best validation Macro-F1 (0.2142) was reached at epoch 13; by epoch 15 the training Macro-F1 had reached 0.8288 while validation Macro-F1 remained at 0.2054 — a large train–validation gap indicating substantial overfitting on the 2,176-image fine-tuning set, consistent with prior work showing document classifiers require far larger samples per class to generalize (Harley et al., 2015).

Table 9. *RVL-CDIP Document-Type Classification Summary.*

Metric	Value
Stratified sample size	3,200 images (200/class)
Train / Valid / Test split	2,176 / 384 / 640
Training mode	Full fine-tune (backbone + graph encoder + doc-typed head)
Best Validation Macro-F1	0.2142 (epoch 13)
Final Training Macro-F1 (epoch 15)	0.8288
Test Macro-Precision	0.2194
Test Macro-Recall	0.2906
Test Macro-F1	0.22
Test Accuracy	0.29

Per-class results (Table 10) show a bimodal pattern: five of sixteen classes (budget, file_folder, form, news_article, scientific_report) achieved zero F1 despite balanced 40-sample test support for every class, while five others (email, handwritten, resume, scientific_publication, specification) reached F1 between 0.39 and 0.57. Inspection of the confusion matrix shows the failed classes are not distributed randomly across predictions: a disproportionate share of budget, invoice, questionnaire, and scientific_report instances were predicted as handwritten, and a disproportionate share of file_folder instances were predicted as email. We interpret this as the model latching onto a small number of dominant visual/textual patterns — plausibly triggered by sparse or garbled OCR output on administrative scans — rather than learning discriminative features for the full 16-way task at this sample size.

Table 10. *Per-Class Performance (RVL-CDIP Test Set).*

Class	Precision	Recall	F1-Score	Support
Advertisement	0.20	0.45	0.27	40
Budget	0.00	0.00	0.00	40
Email	0.25	0.88	0.50	40
File folder	0.00	0.00	0.00	40
Form	0.00	0.00	0.00	40
Handwritten	0.29	0.78	0.42	40
Invoice	0.19	0.30	0.24	40
Letter	0.12	0.12	0.12	40
Memo	0.25	0.38	0.30	40
News article	0.00	0.00	0.00	40

Class	Precision	Recall	F1-Score	Support
Presentation	0.27	0.10	0.15	40
Questionnaire	0.60	0.07	0.13	40
Resume	0.46	0.75	0.57	40
Scientific publication	0.41	0.38	0.39	40
Scientific report	0.00	0.00	0.00	40
Specification	0.38	0.45	0.41	40
Macro Avg	0.22	0.29	0.22	640
Weighted Avg	0.22	0.29	0.22	640

This result stands in sharp contrast to literature-reported RVL-CDIP performance for other architectures — for example, Donut's reported 95.3% accuracy (Kim et al., 2022), already cited in Section 5.1 as context for baseline comparison — though that figure comes from training on the full ≈320,000-image RVL-CDIP training partition, roughly 150× the stratified sample used here, under a dedicated document-classification objective rather than a repurposed auxiliary head. We do not read our RVL-CDIP result as evidence against the graph-based reasoning contribution established in Section 5.2, which was isolated through controlled ablation on DocBank; rather, it shows that the document-type classification module specifically — validated in prior sections only on one class per dataset — does not yet generalize to fine-grained, small-sample, real-world classification, and we treat this as a genuine limitation rather than omit it.

6. Discussion

6.1. Central Finding: Graph-Based Reasoning Is Critical for Document Parsing

Our results provide evidence that explicit graph-based relational reasoning improves fine-grained document parsing, though this is based on single-run experiments and should be confirmed with repeated trials.

The graph encoder contributed the largest single performance gain (+0.04 Macro-F1), confirming that sequential tokenization and 2D positional embeddings alone are insufficient for capturing complex document structures.

This finding has both theoretical and practical implications:

Theoretical: Document understanding is fundamentally a relational task. The meaning of a document element depends not only on its content but also on its relationship to other elements. Graph neural networks provide a natural framework for modeling these relationships, enabling more sophisticated reasoning about document structure.

Practical: For applications requiring fine-grained extraction—such as automated form processing, invoice parsing, or scientific literature mining—explicit relationship modeling is essential for handling structurally complex elements like tables, multi-column layouts, and nested hierarchies.

6.2. Cross-Domain Generalization: Transfer Learning from Scientific Papers to Forms

The FUNSD experiment (119 training samples) achieved comparable performance (Macro-F1 = 0.7450) to DocBank with 4,000 samples (Macro-F1 = 0.7435). This suggests that IntelliDocFormer can effectively adapt to new document types with limited data, likely due to:

1) Shared structural priors: Both scientific papers and forms contain hierarchical structures (sections, headers, body text)
2) Graph-based flexibility: The graph encoder captures spatial relationships that transfer across document types
3) Document-type adaptation: The classification module enables domain-specific processing

We note that our FUNSD Micro-F1 (78.98%) and Macro-F1 (74.50%) do not clearly exceed the 79.3% F1 reported for LayoutLM on form understanding [19]. We caution that these figures may not be directly comparable: LayoutLM's reported metric is not explicitly specified as token-level Macro/Micro-F1, whereas our results use the token-level definitions. An exact metric alignment could be verified by re-running LayoutLM under the same evaluation protocol used in this study, enabling a direct comparison under identical experimental conditions. We do not claim our approach surpasses LayoutLM on FUNSD on the evidence presented here; our contribution claim rests instead on the ablation results isolating the graph encoder's contribution within our own architecture.

We also note a simpler, competing explanation for the FUNSD result: FUNSD's 4-class label scheme (question, answer, header, other) is inherently less complex than DocBank's 13-class scheme, which may account for some of the observed sample efficiency independent of architectural

transfer. Disentangling task-complexity effects from genuine cross-domain transfer would require evaluating FUNSD-style low-data performance on a held-out DocBank subset restricted to a comparably small label set - an experiment we leave to future work. This finding is significant for real-world applications where labeled data for each new document type is scarce and expensive to obtain. We additionally evaluated generalization to a third, more heterogeneous domain by repurposing.

IntelliDocFormer's document-type classification head for 16-way classification on RVL-CDIP. The resulting Macro-F1 of 0.22 was substantially weaker than either primary result and far below literature-reported RVL-CDIP figures for other architectures, indicating that the document-type module's current design does not yet scale to fine-grained, small-sample, real-world classification. We report this finding in full, both because it directly addresses reviewer requests for evaluation on additional public benchmarks and because it more precisely delineates the boundaries of what this paper's evidence supports: the graph-encoder contribution isolated in Section 5.2 rests on controlled DocBank ablations and is not undermined by the RVL-CDIP result, but any claim of document-type generalization must now be qualified accordingly.

6.3. Limitations

Despite strong performance, IntelliDocFormer has limitations:

1) Class imbalance: Rare classes (footer, section) achieve zero F1 with limited training data
2) Computational requirements: 13 GB GPU memory limits its deployment on resource-constrained devices
3) Domain specificity: Validation on benchmark datasets may not generalize to all real-world scenarios
4) Data dependency: Performance scales strongly with data availability, limiting applications with minimal labeled data.
5) Document-type classification does not yet generalize to fine-grained categories: on RVLCDIP's 16-class real-world classification task, the document-type module achieved only Macro-F1 = 0.22 at a 3,200-image stratified sample scale, well below its near-perfect performance on the single-class settings validated on DocBank and FUNSD (Section 5.5). This indicates the module's current design and training protocol do not scale to the number and heterogeneity of categories future deployments would require.

6.4. Future Directions

Class Imbalance Mitigation: Implement focal loss with adjustable γ parameter, class-weighted cross-entropy optimization, and synthetic data generation for rare classes.

Model Compression and Quantization: Explore INT8 quantization (2-4x latency reduction) following the integer-arithmetic-only inference scheme of [8], knowledge distillation to smaller models using the teacher-student framework of [7], and pruning of redundant attention heads. Future deployment-oriented optimization could also investigate integer quantization, following established approaches for efficient integer-arithmetic neural-network inference [9].

Cross-Domain Expansion: Extend evaluation to additional document types including invoices, receipts, legal contracts, and medical records.

Our RVL-CDIP experiment (Section 5.5) represents an initial step toward this direction and surfaces concrete next steps: training on substantially more than 200 images per class, comparing frozen-backbone linear-probe evaluation against full fine-tuning to isolate whether the failure stems from optimization or representation, and replacing the lightweight Tesseract OCR pass used here with a higher-quality pipeline, since sparse or garbled OCR output on administrative scans appeared to drive several of the zero-F1 classes in Table 7.

Multilingual Document Parsing: Collect multilingual datasets, implement multilingual tokenization, and explore cross-lingual transfer learning.

Future research will investigate larger-scale training and additional graph reasoning strategies. Further benchmarking against additional document understanding architectures under a unified evaluation protocol would provide a broader assessment of generalizability.

7. Conclusion

This paper introduced IntelliDocFormer, an enhanced multimodal transformer for fine-grained document parsing that addresses the fundamental limitation of existing models: inadequate modeling of relational dependencies between document elements.

By integrating a Graph Attention Network encoder that explicitly captures sequential and spatial relationships between text tokens, IntelliDocFormer achieves strong performance on both DocBank (Macro-F1 = 0.7435) and FUNSD (Macro-F1 = 0.7450), though direct empirical comparison against prior baselines under matched evaluation protocols remains limited.

Our central finding is that explicit graph-based relational reasoning significantly improves document parsing, contributing the largest performance gain in ablation studies (+0.04 Macro-F1). This validates the hypothesis that document understanding is fundamentally relational—the meaning of an element depends not only on its content but also on its relationships to other elements.

The strong cross-domain performance (transfer learning from scientific papers to forms with comparable results despite 33× less training data) suggests that graph-enhanced transformers can effectively adapt to new document types with limited labeled data. This has important implications for practical applications where labeled data is scarce and expensive to obtain.

Future work will address class imbalance, model compression, and cross-domain expansion. The IntelliDocFormer architecture represents a promising direction for document intelligence, combining the strengths of large-scale pre-trained transformers with explicit relational reasoning through graph neural networks.

Abbreviations

AMP	Automatic Mixed Precision
ANLS	Average Normalized Levenshtein Similarity
BERT	Bidirectional Encoder Representations from Transformers
CORD	Consolidated Receipt Dataset
CPU	Central Processing Unit
FN	False Negative
FP	False Positive
FPN	Feature Pyramid Network(s)
FUNSD	Form Understanding in Noisy Scanned Documents
GAT	Graph Attention Network
GNN	Graph Neural Network
GPU	Graphics Processing Unit
INT8	8-bit Integer (Quantization Precision)
KIE	Key Information Extraction
MLP	Multi-Layer Perceptron
OCR	Optical Character Recognition
PICK	Processing Key Information Extraction from Documents Using Improved Graph Learning-Convolutional Networks
RAM	Random Access Memory
RVL-CDIP	Ryerson Vision Lab Complex Document Information Processing
TP	True Positive
ViT	Vision Transformer
VRAM	Video Random Access Memory

Author Contributions

Gabriel Ihuoma Lilian: Conceptualization, Formal Analysis, Methodology, Software, Writing–original draft

Laud Charles Ochei: Supervision, Validation, Writing–review & editing

Martha Ozohu Musa: Resources, Writing– review & editing

Conflicts of Interest

The authors declare no conflicts of interest.

References

[1] Aghajanyan, A., Yu, L., Conneau, A., Hsu, W.-N., Hambardzumyan, K., Zhang, S., Roller, S., Goyal, N., Levy, O., & Zettlemoyer, L. (2023). Scaling laws for generative mixed-modal language models. Proceedings of ICML, 265-279.

[2] Appalaraju, S., Jasani, B., Kota, B. U., Xie, Y., & Manmatha, R. (2021). DocFormer: End-to-end transformer for document understanding. Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 973-983. https://doi.org/10.48550/arXiv.2106.11539

[3] Beltagy, I., Peters, M. E., & Cohan, A. (2020). Longformer: The long-document transformer. arXiv preprint. https://doi.org/10.48550/arXiv.2004.05150

[4] Devlin, J., Chang, M.-W., Lee, K., & Toutanova, K. (2019). BERT: Pre-training of deep bidirectional transformers for language understanding. Proceedings of NAACL-HLT, 4171-4186. https://doi.org/10.18653/v1/N19-1423

[5] Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., Uszkoreit, J., & Houlsby, N. (2021). An image is worth 16x16 words: Transformers for image recognition at scale. Proceedings of ICLR. https://doi.org/10.48550/arXiv.2010.11929

[6] Huang, Y., Lv, T., Cui, L., Lu, Y., & Wei, F. (2022). LayoutLMv3: Pre-training for document AI with unified text and image masking. Proceedings of the 30th ACM International Conference on Multimedia. https://doi.org/10.1145/3503161.3548112

[7] Harley, A. W., Ummadi, A., & Derpanis, K. G. (2015). Evaluation of deep convolutional nets for document image classification and retrieval. Proceedings of the 13th International Conference onDocument Analysis and Recognition (ICDAR), 991-995. https://doi.org/10.1109/ICDAR.2015.7333910

[8] Hinton, G., Vinyals, O., & Dean, J. (2015). Distilling the knowledge in a neural network. arXiv preprint arXiv:1503.02531. https://doi.org/10.48550/arXiv.1503.02531

[9] Jacob, B., Kligys, S., Chen, B., Zhu, M., Tang, M., Howard, A. G., Adam, H., & Kalenichenko, D. (2018). Quantization and training of neural networks for efficient integer-arithmetic-only inference. Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2704-2713. https://doi.org/10.48550/arXiv.1712.05877

[10] Jaume, G., Ekenel, H. K., & Thiran, J.-P. (2019). FUNSD: A dataset for form understanding in noisy scanned documents. Proceedings of the IEEE International Conference on Document Analysis and Recognition Workshops. https://doi.org/10.48550/arXiv.1905.13538

[11] Kim, G., Hong, T., Yim, M., Nam, J., Park, J., Yim, J., Hwang, W., Yun, S., Han, D., & Park, S. (2022). OCR-free document understanding transformer. Proceedings of ECCV. https://doi.org/10.48550/arXiv.2111.15664

[12] Li, M., Xu, Y., Cui, L., Huang, S., Wei, F., Li, Z., & Zhou, M. (2020). DocBank: A benchmark dataset for document layout analysis. Proceedings of COLING, 1234-1245. https://doi.org/10.18653/v1/2020.coling-main.82

[13] Li, Q., Li, Z., Cai, X., Du, B., & Zhao, H. (2023). Enhancing visually-rich document understanding via layout structure modeling [GraphLayoutLM]. Proceedings of the 31st ACM International Conference on Multimedia (ACM MM), 4513–4523. https://arxiv.org/abs/2308.07777

[14] Gemelli, A., Biswas, S., Civitelli, E., LladÃ³s, J., & Marinai, S. (2023). Doc2Graph: A task-agnostic document understanding framework based on graph neural networks. In Computer Vision - ECCV 2022 Workshops (pp. 329-344). Springer. https://doi.org/10.1007/978-3-031-25069-9_22

[15] Smock, B., Pesala, R., & 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), 4624-4632. https://doi.org/10.48550/arXiv.2110.00061

[16] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, Ł., & Polosukhin, I. (2017). Attention is all you need. Advances in Neural Information Processing Systems, 30. https://doi.org/10.48550/arXiv.1706.03762

[17] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). Graph attention networks. Proceedings of ICLR. https://doi.org/10.48550/arXiv.1710.10903

[18] Wang, D., Raman, N., Sibue, M., Ma, Z., Babkin, P., Kaur, S., Pei, Y., Nourbakhsh, A., & 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. https://doi.org/10.18653/v1/2024.acl-long.463

[19] Xu, Y., Li, M., Cui, L., Huang, S., Wei, F., & Zhou, M. (2020). LayoutLM: Pre-training of text and layout for document image understanding. Proceedings of KDD, 1192-1200. https://doi.org/10.48550/arXiv.1912.13318

[20] Xu, Y., Xu, Y., Lv, T., Cui, L., Wei, F., Wang, G., Lu, Y., Florencio, D., Zhang, C., Che, W., Zhang, M., & Zhou, L. (2021). LayoutLMv2: Multi-modal pre-training for visually-rich document understanding. Proceedings of ACL-IJCNLP, 2579-2591. https://doi.org/10.18653/v1/2021.acl-long.201

[21] Yu, W., Lu, N., Qi, X., Gong, P., & 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. https://doi.org/10.48550/arXiv.2004.07464

[22] Zhang, Q., Wang, B., Huang, V. S.-J., Zhang, J., Wang, Z., Liang, H., He, C., & Zhang, W. (2024). Document parsing unveiled: Techniques, challenges, and prospects for structured information extraction. https://doi.org/10.48550/arXiv.2410.21169