POLITECNICO DI TORINO

Master's Degree in Data Science and Engineering

Master's Degree Thesis

Structural-Semantic Dynamic Graph Learning for Document Visual Question Answering

Supervisors

Prof. Luca CAGLIERO

Phd. Davide NAPOLITANO

Phd. Lorenzo VAIANI

Candidate

HUAN XIAO

APRIL 2025

Summary

With the advancements in Natural Language Processing (NLP) and Computer Vision (CV), Document Visual Question Answering (Document VQA) has become an important research area both in industry and academic.

Visual documents are documents that contain cross-modal elements, such as images, tables, and text. The challenge of visual documents lies in their multi-modal nature, complex structure, and the separation of information by pages.

Traditional document question answering are primarily designed for text-only or image-only inputs, making them ineffective when questions that require both text and visual elements. Even when these modalities are integrated, gaps can remain in how they interact and align. Some models have focused on capturing relations to handle the complex structure of documents, but these approaches are limited to intra-page relationships and rely on static weight aggregation for nodes.

To address these challenges, I propose a framework that utilizes a cross-modal model to extract embeddings, integrates information using a document-level structural-semantic graphs, and employs dynamic weight learning to enhance the aggregation. Using cross-modal embeddings as node features, to enhance semantic relationships, I compute similarity of multi-modal node to construct a semantic graph. To capture document-level structural information, I use logical and spatial relations and connect elements across pages to construct structural graphs. To improve information aggregation, I employ Graph Neural Networks (GNN) with Graph Attention Networks (GAT), which dynamically learn attention scores to assign appropriate weights to neighboring nodes. Through a macro-to-micro model analysis, I selected a global GNN learning architecture that enables the model to simultaneously learn global relationships across both structural and semantic graphs.

Document-level graph and cross-modal nodes preserve the original and completed information of paragraphs and images without splitting, allowing the model to construct a more coherent document representation. Using multiple semantic and structural graphs, the model captures global contextual relationships from different perspectives, improving relational understanding. Additionally, the dynamic GAT weight learning mechanism enhances training flexibility, allowing the model to

adaptively focus on critical information.

Experimental results surpass the baseline, demonstrating the effectiveness of our framework. It is a breakthrough unattainable by traditional single-modality or page-level approaches, establishes a strong foundation for future research in Document VQA.

Table of Contents

List of Tables

List of Figures

Acronyms

AI
artificial intelligence

NLP
natural language processing

CV
computer vision

ViT
Vision Transformer

GNN
graph neural network

GCN
graph convolutional network

GAT
graph attention network

Chapter 1

INTRODUCTION

Natural Language Processing (NLP) is a core field of artificial intelligence, aiming to enable computers to understand, process, and generate natural language. With the advancement of technology, NLP has made significant breakthroughs across various fields, including machine translation, text summarization, sentiment analysis, and information retrieval. Question Answering (QA) is an important area in NLP, focused on building systems that retrieve or generate answers from text or knowledge bases. It is widely used in search engines, virtual assistants, and customer service, as well as in fields like medical diagnosis, legal analysis, and financial forecasting.

In recent years, Computer Vision (CV) also is the popular topic solving the problem about images. As the program of NLP and CV, Visual Question Answering (VQA), the intersection of NLP and CV becomes an interesting research direction. VQA combines image analysis and language processing to answer questions querying or understanding images.

At the same time, document Understanding has acquired greater significance in both academic and industrial settings. Documents are essential carriers of knowledge and information, used in various fields such as academic papers, legal contracts, financial reports, technical manuals, and medical records. The complexity and diversity of document content make document understanding a significant challenge. Whether in academic papers, legal documents, or business reports, the content and structure of documents often contain multi-level semantic information, making it difficult for traditional text processing methods to efficiently and accurately understand the information within them. Therefore, document understanding involves not only processing plain text but also considering the document's structure, format, and cross-modal information, such as images, tables, and charts, which presents new opportunities and challenges for modern document understanding technologies.

For the document understanding, cross-modal information fusion is a key research direction in document understanding, especially when dealing with documents that

include multiple types of information, such as images and text. Many real-world documents contain not only traditional text but also images, tables, graphs, and handwritten annotations, which cannot be effectively understood by traditional text-only or vision-only models. Therefore, how to effectively combine textual information with visual information and other cross-modal data is one of the key problems in document understanding.

Combined Document Understanding and Question Answering, in Document QA research, traditional methods focus more on structured data or basic NLP tasks. However, in document understanding scenarios, the structure of spatial and logital(such as headings, titles, subtitles, paragraphs, footnotes, etc.) and semantic relationships within documents are crucial for extracting the correct answer. Thus, understanding and extracting information in complex documents remains one of the core challenges in improving document QA system performance.

To address these challenges, current solutions primarily include text-based models that leverage textual features for question answering. Most of them used patches of text and the layout information of patches. Vision-based models that process splitted image and layout information to understand document formatting. And cross-modal models that combine both textual and visual features. These methods offer multi-dimensional support for document understanding but still have room for improvement, particularly in better fusion of different modalities and enhancing reasoning capabilities. Also some research extract embeddings of split or objects of document, using similarity or graph learning to retrieve the right answer.

Many related research have found the way that combining deep learning with Graph Neural Networks (GNNs) helps understand document and answer question about document. GNNs excel at capturing relationships between nodes and edges, so they are suitable for handling complex semantic and structural information in documents. Therefore, integrating GNNs with cross-modal information fusion is not only a key direction in current document understanding research but also important for improving document question answering systems. As datasets become more diverse, it is worth exploring the development of more generalized cross-modal document QA systems based on graph structures.

Building on this context and existing research, this paper proposes a novel method: Structural-Semantic Dynamic Graph Learning for Document Visual Question Answering. This approach aims to enhance document understanding by combining cross-modal fusion, integrating textual and visual features along with document structural information. Like most graph-based learning methods, I used Graph Neural Networks (GNNs) to capture structural and semantic information in documents by constructing relational graphs. However, we use dynamic attention weights to further improve reasoning capabilities. Our model effectively combines textual and visual information from documents, dynamically learning and aggregating information based on different contexts. This allows the model to

more accurately understand and infer document content. In complex document environments, leveraging semantic and structural graphs, this method significantly improves the accuracy of information retrieval and reasoning, optimizing the performance of document QA tasks.

In the following sections, we will provide a detailed description of our method, including its mechanisms and workings. We will experiment with different types of graphs and Graph Neural Network (GNN) models, and present experimental results demonstrating the effectiveness and generalization ability of our method, proving its superiority in complex document environments.

Chapter 2

STATE OF THE ART

Our task is Document Visual Question Answering (DVQA), which involves working with multimodal models that enable us to understand both images and text simultaneously. A single model extracts embeddings that represent both image and text, crucial for our task of answering questions about documents that contain both visual and textual information.

NLP is essential to our research on the language understanding. It helps process and understand text in documents. Techniques like Transformers enable the interpretation of sentence structures, meaning, and context, which are crucial for answering questions based on text. With attention mechanisms, Transformers can efficiently handle long-range dependencies, making it easier to extract relevant information from documents.

Computer vision (CV), on the other hand, focuses on visual data like images and layouts information. In our task, models such as ResNet and Vision Transformers (ViT) extract important visual features from images.

To connect text and images, models like CLIP (Contrastive Language-Image Pre-training) is getting more and more popular. They align textual and visual embeddings in a shared space by helping bridge the gap between the two modalities. Allowing the model to link different types of data, so that the text and image can be understand or process together, which is more meaningful and explainable. After that, objects from different modality, such as image and text, can be easily retrieve or utilize on downstream tasks. This is especially useful in DVQA, where both visual and textual information must be understood to answer complex questions accurately.

2.1 Natural Language Processing

Natural Language Processing (NLP) is a subfield of artificial intelligence focused on enabling machines to understand, interpret, and generate human language. Key goals in NLP include text classification, named entity recognition, machine translation, and question answering, among others. NLP techniques can be broadly classified into rule-based approaches, statistical methods, and, more recently, deep learning models such as Recurrent Neural Networks (RNNs)[1] and Transformers[2]. Among these, Transformers have become the cornerstone of modern NLP due to their ability to capture long-range dependencies and contextual information efficiently. In this thesis, we explore how Transformer models, particularly in the context of cross-modal tasks, can be used to understand and process textual data alongside visual inputs.

2.1.1 Transformer

The Transformer model, introduced in the paper Attention Is All You Need [2], changed natural language processing (NLP) by replacing recurrent structures with self-attention mechanisms. Unlike traditional models like Recurrent Neural Networks (RNNs) [1] and Long Short-Term Memory Networks (LSTMs) [3], which process tokens one by one, the Transformer processes all tokens at once. This greatly improves efficiency and solves problems like vanishing gradients and slow computation speeds.

The core component of the Transformer is the attention mechanism, specifically the scaled dot-product attention. Given a set of queries Q, keys K, and values V, the attention scores are computed as:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_k}}\right)V, \tag{2.1}$$

where d_k is the dimension of the key vectors. The softmax function ensures that the attention scores sum to one, allowing the model to focus on the most relevant parts of the input.

Multi-head attention extends this mechanism by applying multiple attention heads in parallel, each learning different aspects of the input representation. This enables the model to capture a richer set of dependencies and contextual relationships.

These models can be categorized into main types: encoder-only models, like BERT, which are designed for tasks such as text classification and question answering, decoder-only models, like GPT, which generate text in a more autoregressive manner, and Text-to-text models, such as T5, treat every task as a text generation

problem, making them highly flexible for a variety of NLP tasks, including question answering, translation, and summarization.

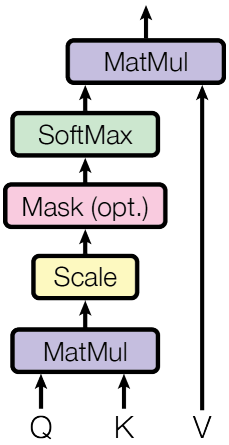

Figure 2.1: Attention score computing

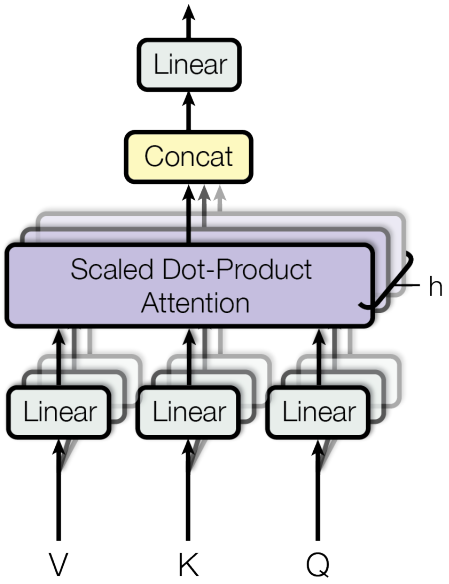

Figure 2.2: Multi-head attention

Encoder-Only Models

Encoder-based models, such as BERT [4] and RoBERTa [5], leverage the Transformer encoder to create contextualized token representations for a given input sequence. These models are particularly effective for tasks requiring deep bidirectional understanding, such as text classification, named entity recognition, and question answering.

The Transformer encoder consists of multiple identical layers, typically six or twelve, depending on the model size. Each encoder layer includes:

- **Multi-Head Self-Attention**: Computes attention scores to model relationships between tokens across the entire input sequence. This mechanism enables the model to capture long-range dependencies more effectively than RNNs.
- **Feed-Forward Network (FFN)**: Applies position-wise transformations to further refine token representations.

- **Residual Connections and Layer Normalization**: Stabilize training and improve gradient flow, preventing the model from overfitting.

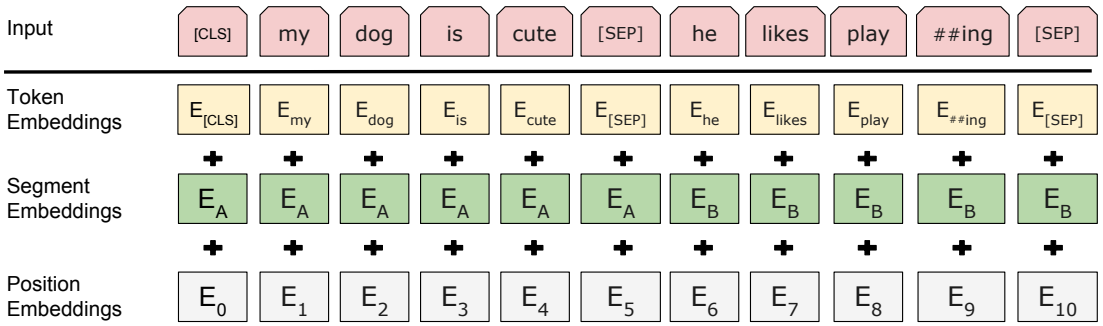

Figure 2.3: BERT[4] input embeddings

BERT[4](full name: Bidirectional Encoder Representations from Transformers), introduces bidirectional attention processing on natural language, and trains on masked language modeling (MLM) and next sentence prediction (NSP) objectives. This bidirectional approach makes the model capture the relationships between words in a sentence above and below. At the same time, positional embeddings added to represent the order of tokens in a sequence, embeddings enable the model to consider the word dependencies based on their relative positions in a sentence.

Additionally, BERT also uses the [CLS] token. It is a special token added at the beginning of each input sequence to represent the whole sequence. After inputting the sentence, BERT not only processes each word as a token but also learns and stores the meaning of the sentence using this special token. During training, for tasks like Next Sentence Prediction (NSP) and text classification, the final hidden state of the [CLS] token is used as a summary of the input. This allows BERT to capture high-level context from the entire sequence. By using bidirectional attention, positional embeddings, and the [CLS] token, BERT performs well on many NLP tasks.

Decoder-Only Models

Decoder-based models, such as GPT and GPT-3 [6], utilize the Transformer decoder to generate text in an autoregressive manner. These models are particularly effective for text generation tasks, including machine translation, dialogue generation, and text completion.

The Transformer decoder, like the encoder, consists of multiple layers, but with additional mechanisms to control the flow of information. Each decoder layer includes:

- **Masked Self-Attention**: Ensures that predictions for a given token do not depend on future tokens by using a causal mask, which prevents information leakage during training.
- **Multi-Head Attention over Encoder Outputs**: Enables the decoder to attend to relevant parts of the input sequence when generating output.
- **Feed-Forward Network (FFN)**: Further processes token representations, similar to the encoder.

GPT-style models remove the encoder entirely and rely solely on the autoregressive decoder, making them highly effective for tasks requiring open-ended text generation. These models are trained using unidirectional language modeling, predicting each token based only on past context.

Encoder-Decoder Models

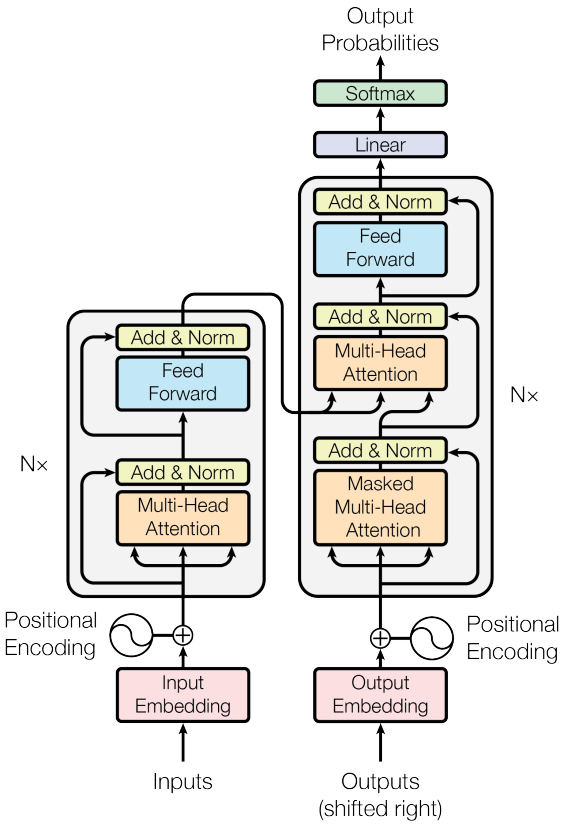

Figure 2.4: The Transformer[2] - model architecture

Encoder-decoder models, such as T5 [7] and BART [8], retain both the encoder and decoder components of the Transformer. These models are particularly useful for sequence-to-sequence tasks, such as machine translation, summarization, and text-to-text transformations.

The encoder processes the input sequence into a latent representation, which the decoder then utilizes to generate the output sequence. The primary differences from the standard Transformer include:

- **Denoising Pretraining (BART)**: BART is trained by corrupting input sequences and learning to reconstruct them, making it robust for text generation and recovery tasks.
- **Unified Text-to-Text Framework (T5)**: T5 reformulates all NLP tasks as text-to-text transformations, enabling a consistent training paradigm across multiple applications.

By integrating bidirectional encoding and autoregressive decoding, these models achieve high performance across diverse NLP benchmarks. Their flexibility makes them highly adaptable for fine-tuning on specific tasks with minimal modifications.

In summary, the Transformer model and its variants have significantly advanced NLP, enabling efficient processing of large-scale text data. Encoder-only models excel in understanding and classification tasks, decoder-only models specialize in text generation, and encoder-decoder models bridge the gap by handling sequence-to-sequence transformations effectively. These advancements continue to shape the future of NLP applications.

2.2 Computer Vision

2.2.1 ResNet in Computer Vision

ResNets, named Residual Networks[9], 'residual' means generally a quantity left over at the end of a process. It solve the vanishing gradient problem in deep networks using residual connections. These shortcut connections allow gradients to flow directly through identity mappings, making it possible to train very deep models.

A standard ResNet consists of several residual blocks, each containing convolutional layers, batch normalization, and ReLU activation. The key innovation of ResNets is the identity shortcut that skips one or more layers by residual connection. Because of that, the information is preserved across deep architectures. Variants like ResNet-50, ResNet-101, and ResNet-152 mainly differ in the number of layers. More layers allow the model to learn more complex features, but they also make the model slower and require more computing power so that can be chose based on the needs.

ResNet has been essential in tasks like image classification, object detection, and segmentation. Modern architectures like EfficientNet and RegNet also build on ResNet's principles, using scaling strategies to improve performance.

2.2.2 Vision Transformer

Vision Transformer (ViT) [10] applies the Transformer architecture to image processing, moving away from CNN-based feature extraction. Instead of using convolutions, ViT divides an image into fixed-size patches, flattens them into vectors, and processes them with a Transformer encoder using positional information, similar to how text is processed.

ViT follows the standard Transformer structure, using self-attention to model global dependencies. Unlike CNNs, ViT directly learns spatial relationships between patches, making it very effective for large datasets. To improve training on smaller datasets, techniques like hybrid ViTs (combining CNNs and Transformers) and self-supervised pretraining have been proposed.

Compared to ResNet, ViT is better at capturing long-range dependencies but requires large amounts of training data to generalize well. Variants like DeiT (Data-efficient ViT) and Swin Transformer improve computational efficiency and adaptability to hierarchical structures.

Both ResNet and ViT are important models in modern computer vision, and hybrid models and attention-based architectures continue to push the boundaries of visual understanding.

2.3 Cross-Modality Models

In cross-modal learning, text and images belong to different modalities, and their underlying feature representations differ significantly. For example, text consists of discrete symbolic sequences, while images are continuous pixel distributions. This modality gap makes it challenging for machines to directly associate and understand semantic information between the two modalities.

To address this issue, semantic alignment is crucial. The core goal of semantic alignment is to project data from different modalities into a shared semantic space, ensuring that text and images with the same meaning are mapped close to each other while unrelated content remains distant. This alignment mechanism benefits several tasks:

- Cross-Modal Retrieval: In tasks like text-to-image retrieval, the model can understand textual semantics and find images that match the given description.
- Multimodal Understanding: The model learns associations between modalities, enabling applications such as image captioning and visual question answering (VQA).

Within this context, Cross-Modality Models provides an efficient approach to semantic alignment

2.3.1 CLIP

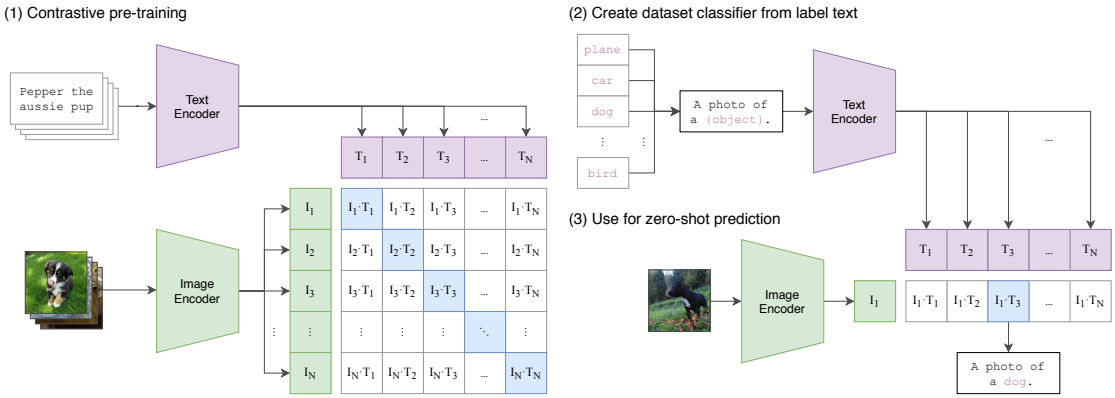

Figure 2.5: Summary of CLIP approach.

CLIP (Contrastive Language–Image Pretraining) [11]is one of the foundational models for cross-modal representation learning. Its core idea is to map images and text into a shared semantic space through contrastive learning shows in Figure2.5.

CLIP employs a dual-tower architecture, using an image encoder and a text encoder to independently encode images and text. The model then optimizes the similarity between the two modalities using a contrastive loss function. Specifically, CLIP maximizes the cosine similarity of matched image-text pairs while minimizing the similarity of unmatched pairs, achieving cross-modal semantic alignment.

CLIP's image encoder uses either a Vision Transformer (ViT) or a ResNet with EfficientNet-style scaling. The ViT encoder adds an extra layer normalization to the patch and position embeddings, while the ResNet encoder uses compute-balanced scaling to improve performance with minimal overhead.

The text encoder is a Transformer [2], modified from GPT-2, with 12 layers, 512 hidden units, and 8 attention heads. It processes BPE-tokenized text (up to 76 tokens), using [SOS] and [EOS] markers, with the final embedding activated by [EOS]. Masked self-attention is used to ensure compatibility with pre-trained models and support future extensions.

A pooler connects the text and image encoders, aligning them to the same dimensionality and mapping them to a shared space.

CLIP is trained with a contrastive objective on 400 million image-text pairs. The model aims to maximize the cosine similarity for matching pairs while minimizing it for 0-labeled pairs. So that it can learn rich associations between vision and language.

The core of CLIP's training is a symmetric contrastive loss function that aligns image and text embeddings in a shared multimodal space. Given a batch of n image-text pairs, CLIP extracts feature representations for each modality: I_f and T_f

Then projected into a joint space using learnable projection matrices W_i and W_t, followed by L2 normalization:

$$I_e = \frac{I_f W_i}{\|I_f W_i\|}$$

$$T_e = \frac{T_f W_t}{\|T_f W_t\|}$$

The cosine similarities are computed and scaled by a learnable parameter τ:

$$\text{logits} = I_e \cdot T_e^T \cdot e^\tau$$

CLIP employs symmetric contrastive loss, treating both image-to-text and text-to-image retrieval. For each image I, the correct text T is treated as the positive class, and vice versa, The final loss is computed as the average of these two terms:

$$\text{loss} = \frac{\text{CELoss}_I + \text{CELoss}_T}{2}$$

This training approach makes CLIP to learn a representation with strong generalization, making it highly effective in zero-shot learning tasks. This means that even without specific domain training, it can also achieve satisfied accuracy. And applicable to various tasks, such as image classification, text retrieval, and cross-modal understanding. Many follow-up cross-modal models have been developed based on CLIP's dual-tower architecture and contrastive learning framework, advancing the field of image-text representation learning. Among these models, Jina CLIP and EVA CLIP have made improvements in training strategies and data.

Jina CLIP[12] uses a **multi-stage training strategy** to enhance vision-language alignment and text representation. This method ensures strong performance in both vision-language tasks and text-only tasks.

EVA-CLIP[13] improves the CLIP framework by building on the EVA[14] and EVA-02[15] vision foundation models, enhancing cross-modal understanding. It inherits the powerful Vision Transformer (ViT) encoder from EVA-02, using large-scale pretraining to improve the generalization of visual representations and optimize computational efficiency for faster training and inference. Compared to the original CLIP, EVA-CLIP integrates improved Feedforward Networks (FFNs) in its dual-tower architecture, improving the nonlinear modeling of text and image features. Additionally, EVA-CLIP uses Mask Image Modeling (MIM) pretraining strategies and multi-scale feature extraction from EVA-02, strengthening the robustness of visual representations and allowing the model to capture finer image details. In terms of training strategies, EVA-CLIP applies semantic alignment loss to refine text-image matching quality, ensuring better consistency between visual and textual embeddings, which improves performance in tasks such as image-text retrieval and cross-modal classification.

2.3.2 BLIP

BLIP, full name is Bootstrapped Language-Image Pre-training[16]. It is a vision-language pre-training (VLP) framework, designed to learn from noisy image-text pairs. Its core model, the multimodal mixture of encoder-decoder (MED), supports both understanding and generation tasks. BLIP uses a Vision Transformer (ViT) as the image encoder and a text encoder similar to BERT. It has three modes: unimodal encoding, image-grounded text encoding, and image-grounded text decoding. The pre-training objectives include: Image-Text Contrastive Loss (ITC) for aligning image and text features, Image-Text Matching Loss (ITM) for fine-grained multimodal representation, and Language Modeling Loss (LM) for text generation. BLIP also introduces CapFilt, a method that generates and filters captions to improve data quality, leading to better performance on downstream tasks.

BLIP-2[17] improves BLIP by using frozen pre-trained unimodal models, making training more efficient and effective. It introduces the Querying Transformer (Q-Former), which connects image and text representations. The pre-training happens in two stages: first, learning vision-language representation using a frozen image encoder, and second, vision-to-language generation using a frozen large language model (LLM). This approach helps BLIP-2 perform well in cross-modal tasks like image-text retrieval and visual question answering.

Compared to CLIP, BLIP and BLIP-2 have different goals and architectures. CLIP focuses on contrastive learning, training an image encoder and a text encoder to align features in a shared space for open-set retrieval tasks. BLIP, in addition to contrastive learning (ITC), includes image-text matching (ITM) and text generation (LM), giving it both understanding and generation abilities. BLIP-2 further enhances cross-modal generation by using a frozen LLM, making it more effective than CLIP in tasks that require converting visual information into text, such as captioning and question answering.

2.3.3 LLaVA

The paper titled *Visual Instruction Tuning* [18] introduces *LLaVA (Large Language and Vision Assistant)*, a multimodal approach that combines a large language model (LLM) with a vision model, aiming to enhance **visual instruction tuning** and enable more natural cross-modal understanding and generation.

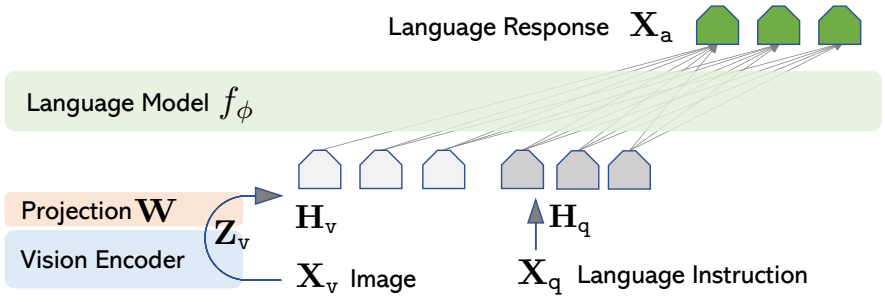

Figure 2.6: LLaVA network architecture

The core architecture of LLaVA consists of a pre-trained vision encoder and a large language model. The vision encoder, CLIP ViT-L/14[11], extracts image features $Z_v = g(X_v)$. To align these visual features with the word embedding space of the LLM, LLaVA applies a trainable projection matrix W, mapping Z_v to language embedding representations as $H_v = W \cdot Z_v$. The language model used is Vicuna, known for its strong instruction-following capabilities among publicly available LLMs.

LLaVA employs a **two-stage instruction tuning** training method to ensure efficient alignment of visual features with the language model and enable end-to-end optimization:

- **Feature Alignment Pre-training** In this stage, only the projection matrix W is trained, while the vision encoder and LLM weights remain frozen. The dataset is derived from CC3M[19] and converted into single-turn conversations, where the input consists of an image X_v and a corresponding question X_q, and the output is the original caption X_a. The goal is to optimize the projection matrix so that the visual features H_v align with the LLM's text embedding space, effectively training a visual tokenizer compatible with the LLM.
- **End-to-End Fine-tuning** In the second stage, LLaVA unfreezes the LLM and optimizes the projection layer while keeping the vision encoder weights unchanged, meaning the trainable parameters are $\theta = \{W, \phi\}$. The goal of this stage is to enhance LLaVA's multi-turn conversation abilities using large-scale instruction-following data. LLaVA is evaluated on the ScienceQA[20] benchmark, which includes detailed reasoning and explanations, allowing the model to perform complex reasoning with multimodal contextual information.

LLaVA achieves efficient cross-modal alignment and enhances vision-language reasoning through instruction tuning. Its lightweight projection layer design enables rapid data-centric experimentation, while the end-to-end fine-tuning approach ensures strong generalization capabilities. LLaVA provides an efficient and flexible solution for multimodal AI tasks, capable of extracting and leveraging cross-modal embeddings for various applications.

For the unification of multimodal embeddings: Inspired by LLaVA and previous text embedding work of 'Scaling text embeddings of Jiang[21]', **E5-V**[22] proposes a prompt-based representation method with MLLMs. The key idea is to explicitly instruct MLLMs to represent multimodal inputs in words. Specifically, E5-V employs structured prompts:

Text prompt:

`<text> Summary of the above sentence in one word:`

Image prompt:

`<image> Summary above image in one word:`

E5-V observes that these prompts effectively remove the modality gap between text and image embeddings, leading to a unified representation space.

For the backbone of E5-V, they use LLaVA-NeXT-8B[18] with a frozen CLIP ViT-L[11] as the visual encoder. The fine-tuning process is applied to the LLM of LLaVA-NeXT-8B, and act on push closer the cross-modality embeddings.

This design not only removes the modality gap but also allows the model to generalize better across different multimodal tasks without requiring additional multimodal training data.

2.4 Graph Neural Networks

With the rapid development of deep learning techniques, traditional neural networks have achieved remarkable success in handling structured data such as images and text. However, much of the data in the real world exists in the form of graphs, such as social networks, molecular structures, and knowledge graphs. To process such graph-structured data, Graph Neural Networks (GNNs)[23] have emerged. GNNs are the product of combining deep learning techniques with graph structures, enabling effective processing and extraction of useful information from graph data.

A Graph Neural Network (GNN)[23] refers to a class of models that apply neural networks to graph-structured data. The core idea of GNNs is to propagate and aggregate node features, through the structural information of the graph, thereby learning vector representations of the graph or its nodes. These vector representations can be used for various downstream tasks, such as node classification, link prediction, and graph classification.

From the perspective of information propagation, GNNs can be mainly divided into the following categories:

2.4.1 Graph Convolutional Network

GCN is one of the most classic models in GNNs. It borrows the idea of Convolutional Neural Networks (CNNs)[24] and aggregates information from neighboring nodes through graph convolution operations (Fig.2.7. The core formula of GCN is as follows:

$$H^{(l+1)} = \sigma(\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}H^{(l)}W^{(l)})$$

Here, $\tilde{A} = A+I$ is the adjacency matrix of the graph with added self-loops, where A is the original adjacency matrix and I is the identity matrix. $\tilde{D}$ is the degree matrix, with diagonal elements $\tilde{D}_{ii} = \sum_j \tilde{A}_{ij}$. $H^{(l)}$ represents the node features at the l-th layer, $W^{(l)}$ is the learnable weight matrix, and σ is the activation function (e.g., ReLU). GCN gradually aggregates information from neighboring nodes through multiple convolution layers to obtain the final node representations.

In GCN, the update of node features is achieved through a weighted average of the features of neighboring nodes. The weights here are determined by the graph's topology, specifically, they are related to the degree of the nodes. The degree matrix $\tilde{D}$ is a diagonal matrix where each diagonal element represents the degree of a node (i.e., the number of neighboring nodes). Through the operation $\tilde{D}^{-\frac{1}{2}}\tilde{A}\tilde{D}^{-\frac{1}{2}}$, GCN normalizes the adjacency matrix, ensuring that each node's feature update depends not only on the features of its neighbors but also on their degrees.

For example, suppose node i has 3 neighbors and node j has 5 neighbors. In GCN, the features of node i's neighbors will be divided by $\sqrt{3}$, while the features

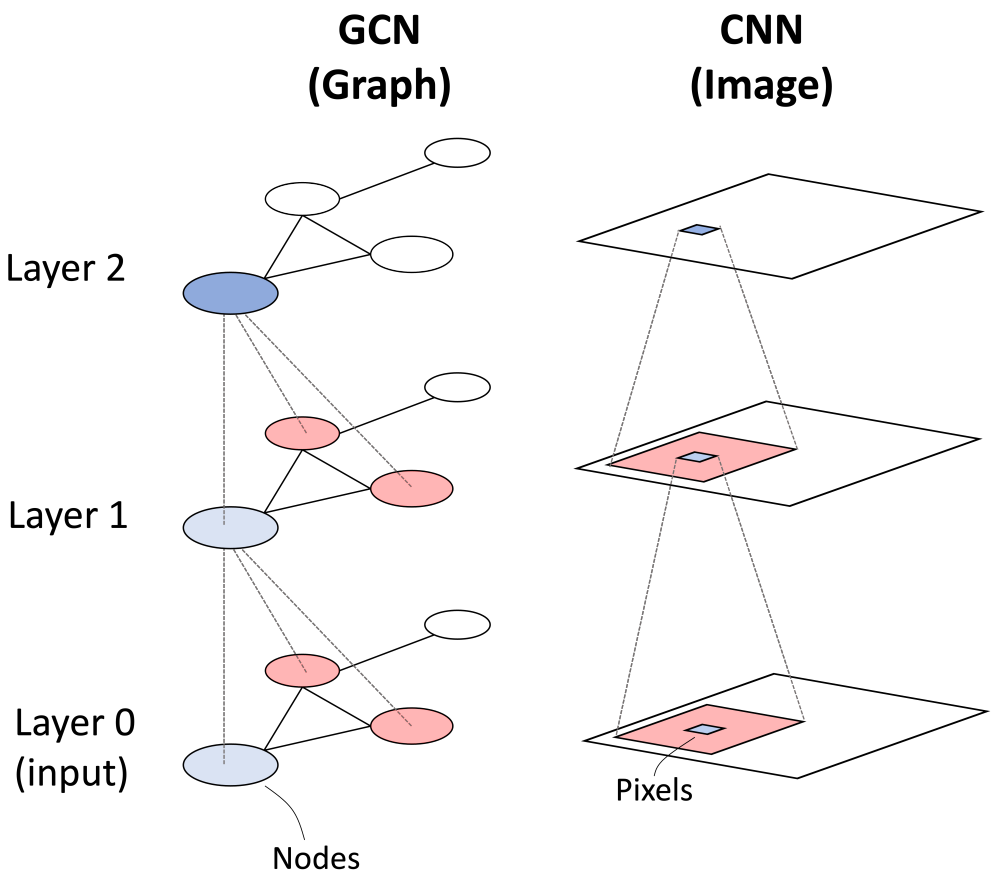

Figure 2.7: Convolution-like operation of GCN[25]

of node j's neighbors will be divided by $\sqrt{5}$. This normalization ensures that nodes with higher degrees do not disproportionately influence the feature propagation process, making the information propagation more balanced.

The advantage of GCN is the ability to capture local structure information of the graph while maintaining computational efficiency. However, the limitation of GCN is that all neighbor of the same node are given the same weight, and it is impossible to distinguish the importance of different neighbors.

2.4.2 Graph Attention Network

GAT introduces attention mechanisms to improve the information propagation process. Unlike GCN, GAT does not simply average the information from neighboring nodes but assigns different weights to each neighbor through attention mechanisms. The core formula of GAT is as follows:

$$h_i^{(l+1)} = \sigma\left(\sum_{j\in\mathcal{N}(i)} \alpha_{ij} W h_j^{(l)}\right)$$

Here, α_{ij} is the attention weight between node i and node j, calculated as (in most of usage and also PyG library):

$$\alpha_{ij} = \frac{\exp(\text{LeakyReLU}(a^T[Wh_i \| Wh_j]))}{\sum_{k\in\mathcal{N}(i)} \exp(\text{LeakyReLU}(a^T[Wh_i \| Wh_k]))}$$

Here, a is a learnable attention vector, $\|$ denotes vector concatenation, $\mathcal{N}(i)$ is the set of neighbors of node i, and W is the learnable weight matrix. GAT can better capture complex relationships between nodes in the graph through attention mechanisms.

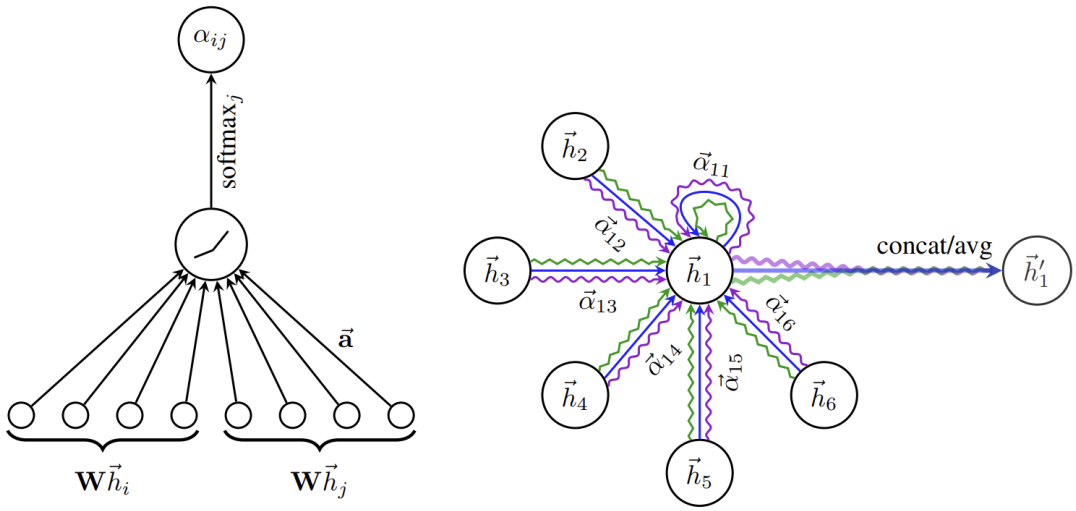

Figure 2.8: Attention mechanism and multi-head attention of GAT[26]

As the left figure of Fig.2.8 he attention mechanism $a(W\tilde{h}_i, W\tilde{h}_j)$ employed by our model, parametrized by a weight vector $\tilde{a} \in \mathbb{R}^{2F'}$, applying an activation.

An illustration shwon in Fig.2.8 of multi-head attention (with $K = 3$ heads) by node 1 on its neighborhood. Different arrow styles and colors denote independent attention computations. The aggregated features from each head are concatenated or averaged to obtain $\tilde{h}'_1$.

In GAT, the attention weights α_{ij} are computed based on node features, rather than being directly determined by the graph's topology as in GCN. This means that GAT can dynamically assign different importance to each neighbor based on their features. For example, in a social network, certain friends may have a greater

influence on a user's behavior, and GAT can assign higher weights to these friends through the attention mechanism.

The advantage of GAT lies in its ability to dynamically assign different importance to different neighbors, making it more flexible in handling heterogeneous graphs (i.e., graphs with diverse node and edge types). Additionally, GAT's computational complexity scales linearly with the size of the graph, making it suitable for large-scale graph data.

Chapter 3

DATASET

This chapter will first introduce the dataset used in my research. The dataset, **PDF-VQA[27]: A New Dataset for Real-World VQA on PDF Documents**, is highly valuable because it provides not only document images but also structured information, making it a rich resource for document understanding tasks.

In this dataset, document elements are segmented into **objects** using bounding boxes. Each object is treated as a fundamental unit for analysis, with its position, category, and logical structure relationships recorded. This structured approach provides a new perspective: the useful information in a document is not limited to just images or raw text. Instead, it includes a wealth of complex details, such as spatial layouts, hierarchical structures, and semantic relationships, all of which can be leveraged to improve understanding and reasoning over documents.

Furthermore, PDF-VQA offers a collection of **question-answer pairs**, which are systematically categorized into different tasks. This task-based division allows researchers to concentrate on specific challenges, such as understanding document layouts, extracting key information, or reasoning about relationships between elements. By providing both structured data and diverse question types, this dataset serves as a powerful benchmark for advancing Visual Question Answering (VQA) in real-world document analysis.

3.1 Dataset

The PDF-VQA dataset [27] focuses on the comprehensive understanding of PDF documents, with data sourced from visually-rich documents in the PubMed Central (PMC) Open Access Subset. Each document file is accompanied by a corresponding XML file that provides structured representations of textual content and graphical components.

In detail, PDF-VQA consists of three modules. The first module processes scanned document images for visual information extraction. Each image represents a scanned version of a document page, capturing its visual features of text, graphics, tables, and images. This module is designed to offer visual information, which is crucial for understanding the document's layout or extracting embeddigns of different elements.

In the second module, document parsing is enhanced, by splitting document elements into objects via bounding boxes. Each object serves as a basic unit of analysis, and its position, category, and logical structure relationship are recorded. To do this, they used the pre-trained Mask R-CNN model to extract bounding boxes and classify document elements. The rich visual and text information is extracted by processing the document page. The results include five main categories: 1) context, 2) title, 3) list, 4) table, and 5) image. They also provide bounding-box coordinates, gaps between elements, parent-child relationships, and extracted text content for each document element. These annotations, structured at the page level, serve as the foundation for subsequent document understanding tasks.

The third module generates question-answer pairs, categorized into three different QA tasks, each focusing on a specific aspect of document understanding.

According to different quesition type, the dataset also consists of three subsets designed to assess different aspects of document understanding:

- Task A: Page-level Document Element Recognition.

  Questions focus on verifying the existence of elements and counting their occurrences, emphasizing spatial understanding. Answers are typically yes/no or numerical values from a fixed set.

- Task B: Page-level Document Layout Structure Understanding.

  Questions require recognizing layout structures and extracting relevant texts. Structural understanding focuses on spatial positioning and reading order, while object recognition involves identifying specific document elements and their logical hierarchy.

- Task C: Full Document-level Understanding.

  This task extends understanding to the entire document, requiring multi-page content retrieval and hierarchical reasoning. Questions involve identifying sections related to specific elements, recognizing parent-child relationships, and extracting high-level summaries of relevant content.

My project aims to address the challenges of Task C in the PDF-VQA dataset, which focuses on full document-level understanding. Unlike Tasks A and B, which operate at the page level, Task C requires reasoning across multiple pages to extract relevant information and establish hierarchical relationships between different document elements. This involves identifying parent-child relationships, linking references across pages, and synthesizing information from multiple sections to provide coherent answers.

In the following dataset introduction, I will focus on providing a detailed explanation of Task C. This will serve as the foundation for strategies aimed at improving document-level understanding.

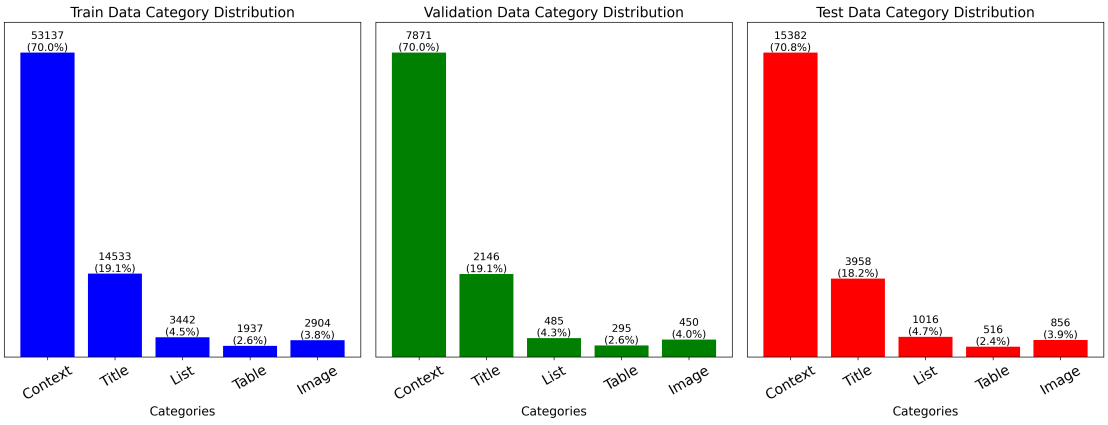

Figure 3.1: Categories Distribution of Task C

To better understand the distribution of the dataset, Figure 3.1 shows the category distribution of the training, validation, and test data. As can be seen from the figure, the Content category accounts for around 70% of the three datasets which is the most category. The table category accounts for 2.6% of the training data and validation data, and 2.4% of the test data, Image around 4% of three dataset. In the future, we will consider mapping different objects into the same space, and the structure of the dataset provides a valuable reference for this task. Specifically, we will explore whether to treat tables as textual or visual data in this unified space.

Task Type	Train	Valid	Test	Total
Document	800	115	232	1,147
Question	3,951	581	1,121	5,653

Table 3.1: Dataset Split for Task C

For Question-answer pairs module, Table 3.1 summarizes the dataset split for Task C. The dataset includes 1,147 documents, with 800 used for training, 115 for validation, and 232 for testing. In total, there are 5,653 question-answer pairs, distributed across the train, validation, and test sets with 3,951, 581, and 1,121 instances, respectively.

3.2 Dataset relation information

Visually rich scientific documents typically follow a layout structure. And they are organized hierarchically, including sections, subsections, tables, figures, and their captions. Understanding these layout structures and hierarchical relationships is important for improving document comprehension. Graph structures have been widely used in various tasks to effectively represent relationships between objects. Inspired by this approach, the PDF-VQA dataset annotates two types of relationships for each document: logical relationships (LR) and spatial relationships (SR). These two structures explicitly represent the logical and spatial relationships between document elements and can be directly utilized by deep learning models to enhance feature representation and document understanding.

SR (Spatial Relationships) describe the spatial arrangement of document elements and contain various types of information. This is based on the absolute positions and bounding box coordinates of elements. For each document element within a single page, we identify eight types of relative spatial relationships with all other elements: top, bottom, left, right, top-left, top-right, bottom-left, and bottom-right. Additionally, we annotate the gaps between all bounding boxes within a page, which refers to the distance between adjacent bounding box edges. Another key annotation is the order list, which encodes the reading order of objects within the page. Since many scientific articles are formatted in two columns, this ordering ensures that the first column is read before the second. These spatial relationships help the model better understand the document's layout structure.

LR (Logical Relationships) focus on the hierarchical structure of the document. It captures parent-child relationships between document elements, representing logical connections between different parts of the document. For example, LR defines relationships between a title and its corresponding content or between a figure and its descriptive text. These hierarchical relationships are annotated within each page to construct the LR graph. When handling multi-page documents, the LR graphs from individual pages are extended and merged to form a complete representation of the document's logical and spatial structure.

However, these relationships are limited to elements within a single page, and cannot establish cross-page associations. Because of that, it is difficult for the model to process information that needs to be understood across pages. To address this limitation, I propose to integrate these relationships at the document level and incorporate their semantic information. By combining spatial position, and semantic relationships into document level, so that the model can gain a deeper understanding of how documents are organized and thus perform better on visual question answering tasks, especially when dealing with complex multi-page documents.

3.3 Dataset QA

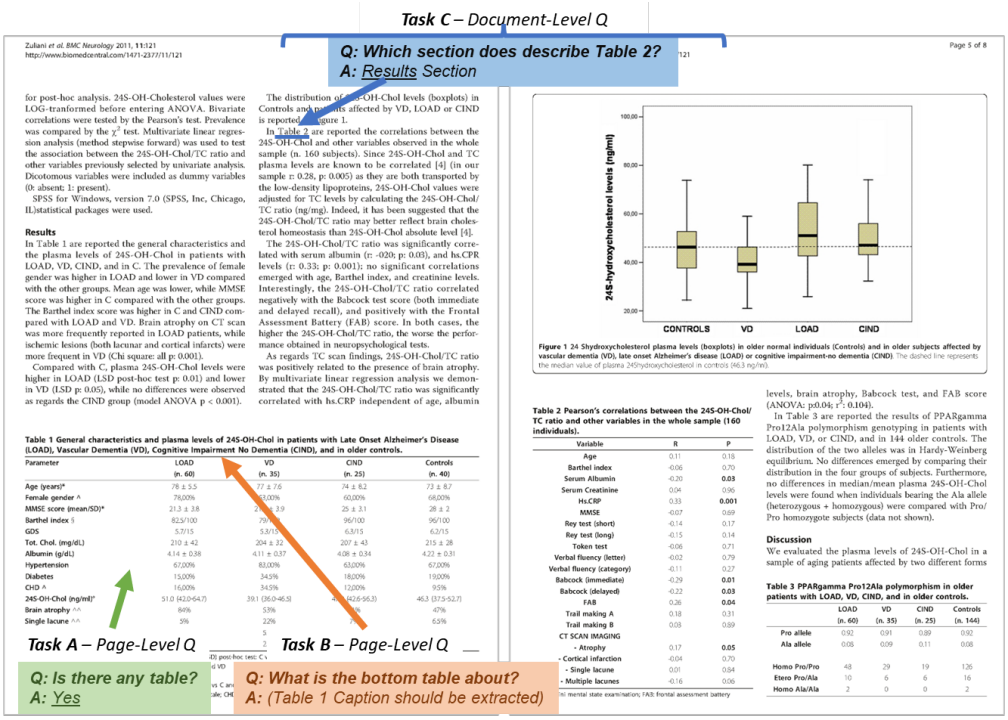

Figure 3.2: PDF-VQA[27] sample questions and document pages for Task A, B, and C.

In the context of Visual Question Answering (VQA), questions are often designed to target specific elements within the document, such as images (Image) and tables (Table). By focusing on these elements, the VQA task challenges the model to integrate both visual and textual information, enhancing its ability to comprehend complex document structures. This approach ensures that the model can handle real-world scenarios where questions often revolve around specific visual or tabular data within a document.

As I discussed, my project will focus on **Full Document-level** visual question answering, the task C of pdf-vqa[27]. This component extends the scope from individual pages to the entire document, aiming to enhance the model's ability to analyze and comprehend document hierarchies. For example, questions such as "Which section describes Table 2?" require the model to locate all sections in the document that describe the table and return the titles of these sections as the answer. This type of question emphasizes the model's ability to understand the overall structure of the document, rather than just the content of individual pages, which is also our main challenge.

Table 3.2: Ratio and exact number of various question types

Question Type	**Percentage**	**Total**
Parent Relationship Understanding	79.71%	4,506
Child Relationship Understanding	20.29%	1,147

In Task C, the ratio and exact number of different question types are presented in Tab.3.2. The questions are categorized into Parent Relationship Understanding and Child Relationship Understanding. Parent Relationship Understanding questions account for 79.71% of the dataset, with a total of 4,506 questions. Child Relationship Understanding questions make up 20.29%, with a total of 1,147 questions.

- **Parent relationship understanding** requires the model to recognize higher-level content related to the query by looking "upward" in the document structure. For example, the model needs to determine which section a specific table or image belongs to, as in the question: "Which section describes Table 1?" Answering such questions involves multi-label retrieval, meaning the model must not only retrieve relevant paragraphs but also identify the corresponding section titles and other related headings.

- **Child relationship understanding**, on the other hand, requires the model to recognize lower-level content by looking "downward" in the hierarchy. This includes extracting all subsection titles or specific fields under a given section. For instance, a question like "What is the Discussion about?" requires the model to locate and answer the content under the "Discussion" section.

Through this design, the component not only expands the scope of document understanding but also showcases the model's ability to process hierarchical structures. In the future, this approach may further enhance the model's capability to comprehend and reason about document layouts at different levels.

They used an automated question and answer generation process to generate questions and answers. Using predefined question templates and document structure information, they generate a large number of diverse question and answer pairs. Each question template contains multiple language patterns to ensure diversity of questions. Furthermore, a functional program is used to automatically generate answers to ensure the accuracy and unify of the answers. For example, for the question "What is discussed in the 'Methods' section?" The functional program generates the answer by extracting all section headings under the "Methods" section.

They also provided an in-depth analysis of the dataset, focusing on the distribution and characteristics of the questions. We will also present statistical insights that highlight key patterns in question formulation and answer types. This dataset

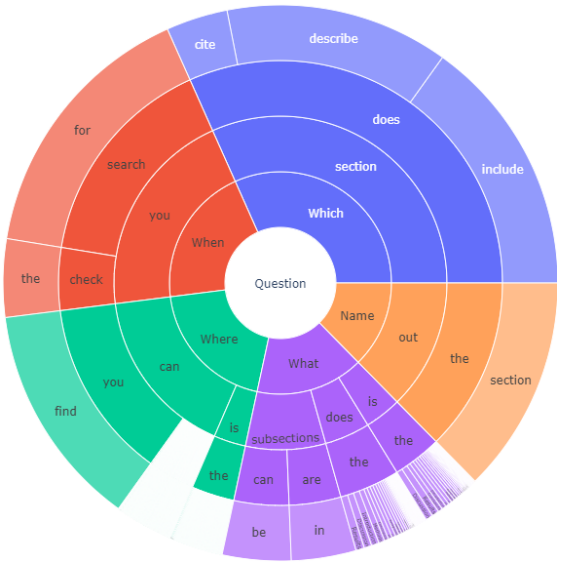

Figure 3.3: The top 4 words of questions

analysis plays a role in understanding the underlying structure and challenges of the task at hand.

The average number of questions per document page in this component (Task C) is 4.93, with an average question length of 15. As shown in the top 4 words frequency plot (Figure 3.3), questions are predominantly distributed around words like "What," "When," "Can you," and "Which." This pattern indicates a focus on understanding the structure of the document rather than specific content, aligning with the hierarchical structure and spatial organization required for this task.

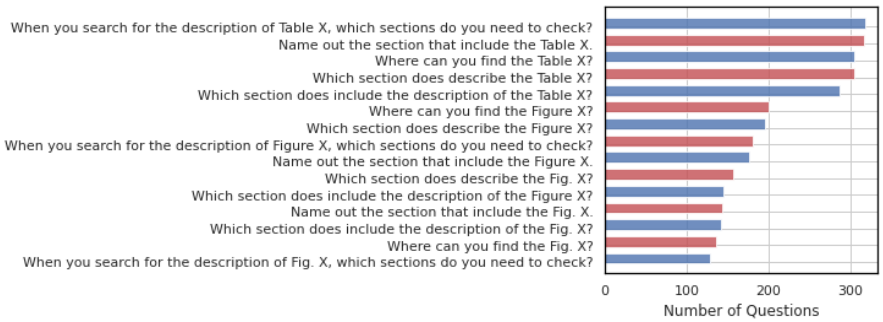

Figure 3.4: Top 15 Frequency Parents Questions

Additionally, 29.38% of the questions in this component are unique, which, although relatively low compared to other document-based VQA datasets, reflects the focus on higher-level document understanding. The questions are designed

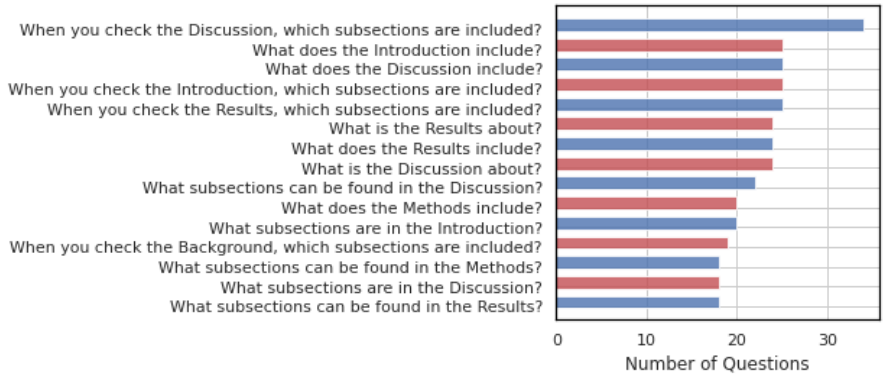

Figure 3.5: Top 15 Frequency Children Questions

to address the document's layout, such as "Name out the section that includes Table X," a style of declarative questioning that encourages models to recognize document structures. This is further supported by the frequency plot of the parent question (Figure 3.4), where common queries related to finding the location of figures and tables (e.g., "Where can you find the Table X?") dominate. The child question frequency plot (Figure 3.5) further illustrates that many questions focus on extracting subsection titles or details within sections, such as "What subsections are in the Introduction?" These question patterns emphasize the need for the model to understand the content at different hierarchical levels, making this component an effective test for document comprehension and layout analysis.

Table 3.3: Rater Agreement for Automatically Generated QA Pairs

Perspective	**Pos(%)**	**Kappa**
Relevance	100	100
Correctness	94.55	80.93
Meaningfulness	99.27	97.34

To evaluate the quality of automatically generated question-answer pairs, ten raters, including deep-learning researchers and crowd-sourcing workers, were invited. Firstly, to determine the relevance between the question and the corresponding page/document, the Relevance criteria were defined. Correspondingly, the Correctness criteria were defined to determine whether the auto-generated answer is correct in relation to the question. In addition, the raters were asked to judge whether the QA pairs were meaningful and possibly appeared in the real world, using the Meaningfulness criteria. After collecting the raters' feedback, the positive rate for each perspective was calculated, and Fleiss Kappa was applied to measure the agreement between multiple raters, as shown in Table 3.3. All three tasks achieved

decent positive rates with substantial or almost perfect agreements. For Task C, both positive rates and agreement across the three perspectives were notable. Furthermore, except for three perspectives, the raters agreed that most of the questions in Task C required cross-page understanding (with a positive rate of 82.91%).

In summary, the document-level parsing results provide a structured approach to document understanding. While they offer valuable spatial and logical relationships, challenges remain in document-level question answering tasks.

Chapter 4

METHODOLOGY

In this chapter, we introduce the challenges we face in our dataset and task, along with our strategies for addressing and improving them. Specifically, we will provide a detailed explanation of our data preprocessing approach, including the integration of cross-modal information, the construction of cross-page relationships, and the extraction of logical structures and document layout information. We will also describe the design of our graph structures, which incorporate text, image, logical, and spatial information to build multiple complementary graphs, such as cross-modal embedding graphs, cross-page text relationship graphs, logical relationship graphs, spatial relationship graphs, and similarity graphs. Based on these graphs, we will design corresponding GNN models and explain how the integration of these structures helps the model effectively learn cross-modal, cross-page, and multi-label relationships. Additionally, we will discuss the incorporation of dynamic attention mechanisms to enhance information aggregation.

Furthermore, we will analyze potential phenomena and challenges that may arise during experiments, including model behavior, possible limitations, and key factors influencing performance. Through this chapter, we aim to present a comprehensive methodology that demonstrates the effectiveness of our approach in the Document Visual Question Answering task.

4.1 Task Defination

I have already introduced my dataset. The challenges I face in Document Visual Question Answering on my dataset can be categorized into four key aspects:

- *Cross-modal challenges*: These arise due to the presence of both textual and visual information in the dataset. The model needs to integrate and reason over multiple modalities effectively.

- *Cross-page dependencies*: The answer to a question may not always be found on the same page as the question itself, requiring the model to track information across multiple pages.

- *Multi-label relationships influenced by logical structures*: The correct answer is not always limited to the explicitly mentioned text in the question. Related elements, such as section headings, may also be valid answers.

- *The impact of document semantics and layout*: Both textual meaning and spatial positioning play roles in determining the relevance of an answer.

Among the many possible strategies of Document Visual Question Answering of PDF-VQA[27], methods such as their LoSpa[27] and research doc-GCN[28] have provided invaluable insights by demonstrating the effectiveness of graph-based models and Graph Neural Networks (GNNs) in aggregating information. These approaches prove that constructing graphs to model relationships among different entities is a powerful means for data processing and analysis.

Furthermore, we draw inspiration from knowledge-graph-based question answering systems, such as QAGNN[29]. Although we do not directly incorporate external knowledge graphs, QAGNN's use of Graph Attention Networks (GAT)[26]—with its dynamic attention weight mechanism that enables the network to focus on the most critical parts of the graph—offers a significant reference for our task. Moreover, GraphDoc[30] also demonstrates the rationality of applying graph-based models and GAT at the document level, albeit its approach is limited to a single modality and a single graph. This dynamic attention mechanism motivates us to consider applying GAT layers at the document level to better capture the interrelationships among various pieces of information.

According to the task challenges and related works, my approach, "Structural-Semantic Dynamic Graph Learning for Document Visual Question Answering," systematically addresses these issues:

- For cross-modal challenges, we construct cross-modal embeddings to unify textual and visual features into a shared representation space.

- For cross-page dependencies, we explicitly model text relationships by linking content across different pages.
- For multi-label relationships and document structure influences, we introduce three types of document-level graphs.

And one more worthable improvement of GNN of document graph:

- Graph Attention Networks (GAT)[26]—with its dynamic attention weight apply on document graphs.

My task addressed the cross-modal challenge, requiring integration of features from both images and text. To achieve this, we design node features that capture semantic information from both modalities by using cross-modal embedding model, facilitating better multimodal understanding. Embeddings can be the node feature and can also compute the similarity of semantic.

My approach include designing and selecting optimal node and edge features to build multiple graph structures. Node features represent cross-modal content. Node feature extend connectivity from the page level to the document level. Edge features capture a variety of relationships, including logical connections, spatial relationships, and similarity measures. All features are at the document level, ensuring that the model can extract relevant information across pages.

We will also analyze different GNN architectures to determine the most suitable model structure, exploring configurations with GAT layers and evaluating their ability to aggregate and process information. Through systematic analysis and experimentation, we aim to fine-tune the model for effective representation of semantic and relational information.

4.2 Loss function

Due to the nature of our task, where a single question may have multiple correct answers (multi-label classification), our model produces continuous scores instead of discrete labels. To effectively handle this, we use BCEWithLogitsLoss (Binary Cross-Entropy with Logits Loss), which is particularly suited for multi-label problems. Since each label is treated independently, this loss function enables the model to assign probabilities to multiple correct answers rather than forcing a single categorical choice.

The function is shown below:

$$\mathcal{L} = -\frac{1}{N}\sum_{i=1}^{N}\left[y_i \log \sigma(x_i) + (1 - y_i)\log(1 - \sigma(x_i))\right]$$

Where:

- x_i represents the raw model output (logits).
- y_i is the target label, which takes values of 0 or 1.
- $\sigma(x_i)$ is the Sigmoid activation function that converts logits into probabilities:

$$\sigma(x_i) = \frac{1}{1 + e^{-x_i}}$$

- N is the total number of samples.

BCEWithLogitsLoss integrates two key steps—applying the sigmoid activation and computing binary cross-entropy—into a single, numerically stable operation. This prevents potential instability that could arise from separately applying the sigmoid function before loss computation.

The main advantage of BCEWithLogitsLoss is the ability to handle multiple correct labels. Select all right answers for each question. Unlike traditional multi-class losses, which assume there is only one correct answer, BCEWithLogitsLoss evaluates each label independently, allowing for more flexible and accurate learning. The final loss can be averaged or summed between labels, ensuring that each incorrect prediction gets the appropriate feedback.

In addition, BCEWithLogitsLoss is widely used in deep learning because of its stability and efficient gradient calculation. By using this loss function, our model is able to efficiently learn from data with non-unique correct answers.

4.3 Graph Preprocessing

4.3.1 Embeddings extraction

From the task and dataset, I can observe that the primary challenges in solving these problems lie in the **cross-modal** nature of the tasks and the **inter-page relationships** within the document information.

In most document research, text is typically chunked, or PDFs are treated as images and cut into pieces, with text and images separately undergoing patch spatial embedding.

However, since our task is to construct a graph capable of understanding document content, including both images and text, we need to place text and images in the same space for relational analysis.

To address these challenges, we no longer rely on single-modality models or simply combine single-modality features, but instead, we utilize cross-modal models. These models reduce the distance between embeddings from different modalities, allowing them to exist in a shared space where semantically similar text and images are brought closer together. This approach helps avoid potential biases that could occur when pooling or combining different modalities in a conventional manner, and it enables the direct use of cross-modal node features, making it possible to build a cross-modal graph for integrated and collaborative understanding of multimodal information.

Therefore, our first goal focuses on the extraction of cross-modal embeddings. We use the object-level text and image representations from PDF-VQA, where the text refers to parsed text and the images and tables are extracted using their bounding boxes, cut from the scanned document pages. It is important to note that we cut out the complete images of the objects. After passing through the cross-modal model, the text and image features are used to extract embeddings, which capture information from both the entire paragraph and the image.

We plan to test different cross-modal models to evaluate their suitability for our task. We aim to select a model that has been pretrained on charts and tables, as it should be well-suited to understanding and processing our document.

To evaluate whether cross-modal models can effectively map images and text into a unified embedding space, I selected several visually similar images from the dataset. Some of these images originate from the same file but differ in key descriptive details. Descriptions of the images were generated using large language models and then manually refined to better reflect the image content. I tested multiple models, including CLIP[11], JINA-CLIP[12], EVA-CLIP[13], BLIP[16] extractor from lavis[31], E5-V[22], and E5-V (table:text), to extract embeddings by integrating these descriptions with the visual content. And will discuss in Experiment and Result part.

4.3.2 Document-level relation

When performing all of the information connections, we re-indexed the local IDs of the objects within the document. Instead of using page-specific IDs, we re-sequenced them across pages. Consequently, all relationships between objects also reflect the new sequential numbering. This ensures that all objects within the document can be connected to one another, even across different pages.

Semantical relation

The advantage of obtaining cross-modal embeddings is that it not only presents the **text or image**, but also allows different modalities to be semantically linked. These **links** are weighted according to their similarity values, so when building a graph, semantically similar objects will be connected to each other. In other words, even objects from different modalities, if they describe similar content, will be linked by a higher similarity value.

First, the connections in the cross-modal graph preserve the integrity of the graph information, enabling the establishment of meaningful relationships between all document objects. By linking objects across modalities based on their semantic similarity, we ensure that both textual and visual elements contribute to the overall understanding. This approach reduces the risk of missing critical relationships, as it allows for richer, context-aware connections between different document components, whether text or image.

Moreover, this cross-modal connection enables the model to capture more complex interdependencies between text and images, improving tasks like visual question answering and document retrieval, where both modalities need to be interpreted together.

Importantly, all similarity calculations and connections are performed at the document level. This means that, regardless of how far apart two objects are or whether they are on different pages, as long as their semantic content is similar, they will be connected through the similarity value.

Structural relation

For document structural relation, I will optimize with respected to the logical information and spatial information from PDF-VQA.

- **Logical relation**: PDF-VQA already provides structural logical relationships, which are described by children and parents. These relationships are crucial as they help understand the overall structural and hierarchical connections, aiding the model in learning to answer questions related to these relationships. However, these relationships are confined to within a single page. Since our task

involves the entire document, we need to extend these logical connections across pages. By linking the relationships logically, we ensure that the information can be understood in the context of the entire document, rather than being limited to individual pages. This approach allows us to build a more comprehensive and coherent representation of the document's structure, facilitating better task performance across the entire document.

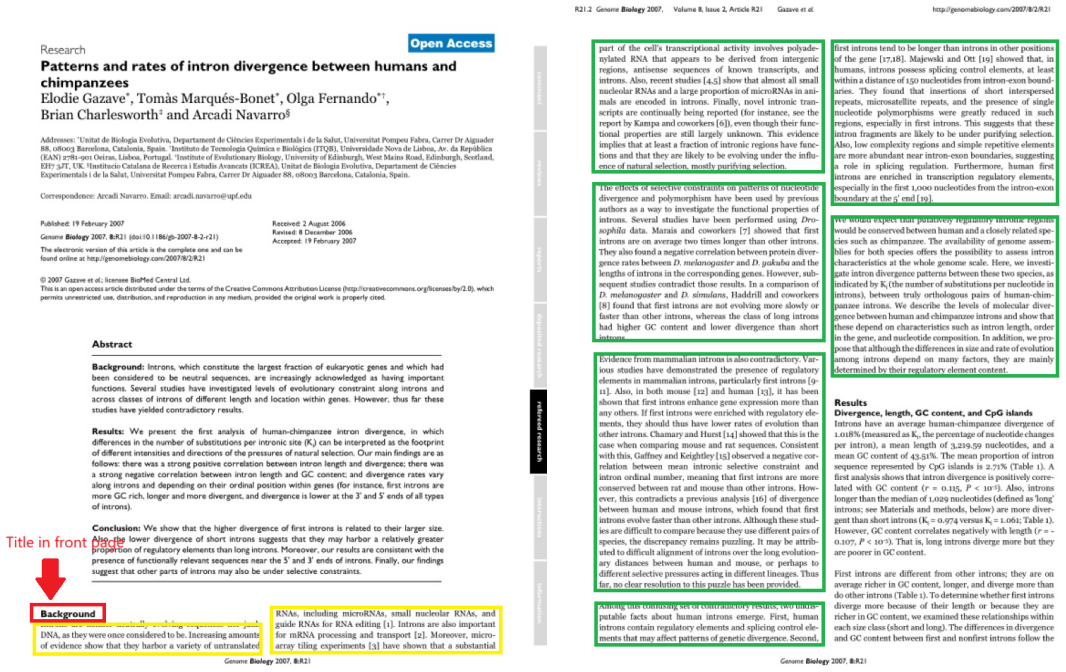

Figure 4.1: Parents-children document leveled relation for adjacent two pages

As shown in the figure4.1, the first page contains an object that acts as a parent, which is **'background'**, and its category is **title**. On the second page, the object within the green box does not have a parent-child relationship, and the children of the 'background' are only the objects within the yellow box. However, when we consider the entire document, the object within the green box should also be considered a child of the 'background', as it belongs to the same semantic group. We retain the object from the first page as the parent and connect it to the context on the second page, continuing this relationship until we encounter the next parent, which is **'Result'**.

This approach allows us to link objects across pages, even when they are separated by page breaks, thus maintaining the structural coherence of the document.

- **Spatial relation**: As related works, Lospa[27] spatial relation follows Doc-GCN[28] by using the visual features of document elements as node representations and the distance to the **two nearest** document elements to weight the edge value. Doc-GCN utilized node features such as visual and density features of each segment when employing spatial relations. However, for our document-level relations, it is challenging to calculate the **cross-page gap**. Upon further analysis, we discovered that when calculating the gap, the nearest two elements could be either horizontally or vertically positioned. This approach lacks reasoning, as in a dual-column document, the horizontally closest elements do not necessarily have a meaningful connection.

  Fortunately, PDF-VQA comes with an ordered list that has a specific reading order. This implies that, on a single page, the object positioned at the lower left will be linked to the object at the upper right as the closest object. This system follows the logical reading order which ensures that the relations between the objects are in accordance with the reading flow of the document. By using this order, the structure of the document can be preserved which makes it easier for the model to interpret and process it.

  This strategy is quite beneficial when applying intra-page information to inter-page relationships. Since the objects on each page are already linked according to a logical order, the ordered lists on each page can be joined throughout the entire document. This not only preserves the structure of the document, but also enhances the understanding of inter-page relationships. In doing this, we make sure that objects on adjacent pages are logically accessible and are able to enhance performance for cross-page queries.

4.4 Graph making

After completing the graph preprocessing, I obtained embeddings for both images and text, which can serve as node features in the graph.

Various types of relationships were identified between nodes, and these relationships can be leveraged as edge features. Logical and spatial relations are used as edge features with a constant value of 1 to indicate the connection between nodes. For the similarity relationships, we use the similarity values themselves as edge features, where high-similarity pairs are connected as neighbors with the similarity value acting as the edge weight. This approach allows us to dynamically construct graphs that connect images and text based on different types of relationships, forming multiple distinct graphs.

To refine the similarity graph, we adopt **top-k similarity** as a selection criterion. In this case, each node is connected to its top-k most similar neighbors, and this connection is also filtered at the document level to ensure context relevance. Rather than setting a fixed threshold for similarity, which can vary significantly across models, the top-k method helps avoid the challenges of defining a narrow threshold. If the threshold is too strict, it may lead to imbalanced connectivity between nodes. By using top-k similarity, we maintain consistent semantic connections between objects while providing flexibility for downstream tasks. In addition, this approach enhances the transferability between tasks, since the similarity values can be used as edge weights. The k value is treated as a hyperparameter that is experimentally optimized to achieve the best performance in different contexts.

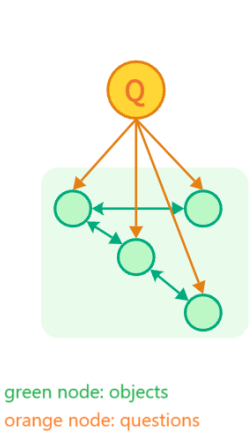

Figure 4.2: Each question connected to object in Graph

As we can see in Fig.4.2, I also connect the questions and objects. This not only helps in understanding the relationship between the question and the objects

during the judgment process but also aids in comprehending the question in the context of all aggregated information after the aggregation step. By connecting the question with the objects, we ensure that the question is integrated into the overall context, facilitating better understanding and more accurate reasoning based on the aggregated data. This approach strengthens the contextual connection between the question and the entire set of relevant information, enabling a more holistic interpretation.

In summary, we have successfully created multiple complete graphs by connecting nodes, objects, and questions under different relationships. This process enables us to theoretically establish three distinct graphs.

In practical applications, we utilize the PyG DataLoader. Unlike the standard PyTorch DataLoader, which processes independent samples, the PyG DataLoader is specifically designed for graph-structured data. It supports batches of different sizes, which means it can efficiently process graphs with different numbers of nodes and edges. This is important for our dataset and graph work where the number of nodes in different documents varies. So by taking advantage of PyG, we can deal with graphs of variable size.

The PyG DataLoader merges adjacency matrices while maintaining batch indices, allowing multiple small graphs to be combined into a single large graph. This enables parallel computation while preserving the topology of each individual graph.

In PyG, the adjacency matrix is typically represented by the `edge_index` tensor, which encodes graph connectivity by listing node pairs that form edges. Specifically, each column of `edge_index`, $[i, j]$, represents a directed edge from node i to node j. Additionally, the `batch` tensor tracks the original graph to which each node belongs, ensuring correct indexing during message passing in GNN layers.

4.5 Model Designing

We already have complete graphs with predefined nodes and edges, as shown in the figure4.3. Our task now is to analyze them using GNN.

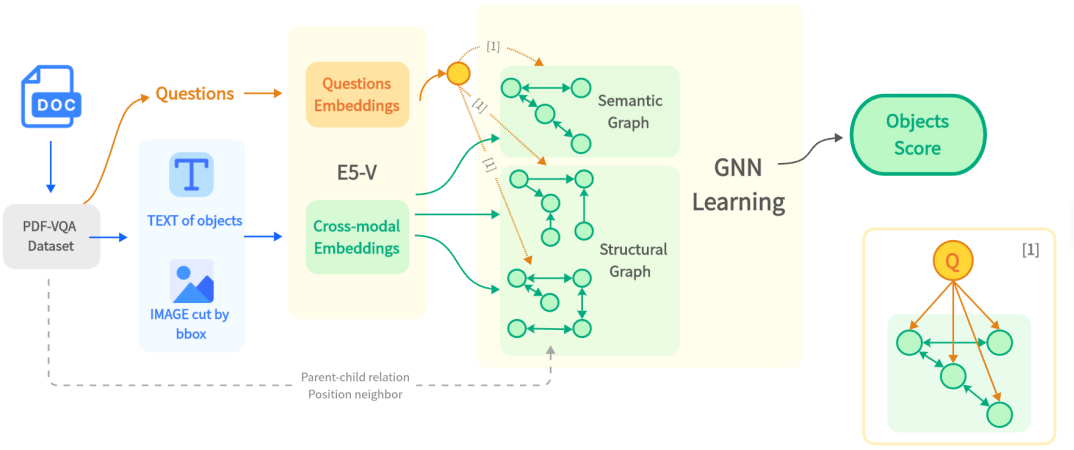

Figure 4.3: Overall Model and Pipeline

For this module, I will focus on the internal structure of the model. Starting from the broader level, I consider the composition of different GNNs. We can also describe it as 'shaping the overall model'. Then, I move to the internal design of the GNN itself, where I can adjust the number of GAT layers. Finally, I improve the GAT structure by tuning the number of attention heads. This process allows us to adjust hyperparameters **from a macro to a micro** level, helping us find the most suitable configuration for our approach.

4.5.1 GNN Composition

The first and most fundamental step in our study is to explore how different graphs can leverage GNNs for information aggregation. Since our dataset contains multiple interrelated graphs, it is crucial to determine the most effective way to capture and propagate information. We investigate two distinct graph-based learning strategies, each with its own advantages and trade-offs:

- **Global Graph Learning**: This approach utilizes a single GNN to process multiple graphs in a unified manner. By integrating information from different graphs, it enables cross-graph feature fusion while reducing parameter overhead. The global model treats all graphs as a shared representation space,

allowing for smoother information flow across different structural components. This strategy is particularly effective for complementary graph pairs (e.g., relation-similarity graphs), where coordinated feature propagation facilitates meaningful pattern discovery. Using a single GNN for integration also improves computational efficiency and avoids redundant processing of shared features. However, this approach requires careful design to ensure that different graph structures contribute effectively without causing information overload. To optimize this learning strategy, we will perform experiments with various combinations of graphs.

- **Combined Graph Learning**: In this approach, different graphs are handled by separate GNNs, allowing each model to preserve graph-specific features and process information independently. This design provides greater interpretability, as each graph has a dedicated feature processor focusing on its unique structure and information flow. By maintaining independent processing pipelines, this method ensures that the distinct properties of each graph type remain intact. However, parallel GNN streams introduce challenges, particularly in combining the outputs of multiple independent networks. If the fusion method is not well-designed, conflicting signals may arise, misaligning the graph representations and weakening feature aggregation effectiveness. To address this, we will use trainable parameters to dynamically adjust the integration of different graphs, ensuring effective combination and information propagation.

By comparing these two methods, we aim to understand the trade-offs between efficiency and interpretability in multi-graph learning scenarios. The global approach offers a more compact and computationally efficient representation, making it well-suited for tasks requiring strong cross-graph interactions. In contrast, the combined approach enhances transparency and specialized processing, improving interpretability at the potential cost of increased computational complexity. In our specific graph learning task, finding the optimal balance between these two strategies is crucial for achieving the best performance. We will conduct systematic experiments to evaluate both methods and determine the most effective approach.

4.5.2 GNN Structure

We have already discussed different strategies for constructing graph combinations and have identified a suitable approach. Now, we are further optimizing the internal design of the GNN, which is critical to the overall performance of the model.

In models such as Doc-GCN and LoSpa, graph neural networks primarily rely on GCN (Graph Convolutional Networks). GCN employs a neighborhood aggregation approach, where each node updates its representation by aggregating

its neighbors' features using a fixed weighted sum. While this method effectively captures local connectivity, it has certain limitations. First, GCN assigns the same weight to all neighbors, leading to a lack of differentiation in information propagation. Additionally, GCN struggles to model complex relationships where different neighbors contribute varying levels of importance to the target node.

To overcome these issues, we adopt GAT (Graph Attention Networks) for aggregation. GAT introduces an attention mechanism that dynamically assigns different weights to neighbors rather than averaging them uniformly. This allows the model to learn the importance of each edge, enabling it to focus more on key information while reducing the impact of irrelevant or noisy neighbors. This selective aggregation makes GAT more flexible in feature representation, particularly for tasks that require distinguishing between different types of relationships. Moreover, GAT is advantageous for heterogeneous graphs, as it effectively models complex structures with diverse dependency types—aligning well with our multi-graph learning task.

Another crucial factor in our model design is the choice of the number of GAT layers. In many graph-based question-answering (QA) tasks, the depth of the GNN directly impacts the model's performance. The number of GAT layers determines the extent to which information propagates through the graph, affecting the richness of node features.

- A shallow GAT (single layer) captures only local neighborhood information, making it suitable for tasks where features primarily depend on first-order (directly connected) neighbors. However, it may struggle to model multi-hop relationships, limiting its ability to capture long-range dependencies.

- A deeper GAT (multi layers) enables information to propagate across multiple hops, allowing nodes to aggregate distant contextual information. This is particularly beneficial for tasks that require reasoning over indirect relationships, such as long-range dependency modeling or cross-document information integration.

However, increasing the number of GAT layers introduces challenges such as over-smoothing and vanishing gradients. Over-smoothing occurs when a deep GNN causes node embeddings to become indistinguishable, reducing the model's discriminative ability. Additionally, deeper networks require more computational resources and may lead to unstable training.

In our research, we systematically analyze the impact of different GAT layer depths on the model, aiming to strike an optimal balance between representational power and computational efficiency. By experimenting with various configurations, we seek to determine the ideal number of GAT layers that maximizes performance while mitigating over-smoothing.

4.5.3 GAT Analysis

Graph Attention Networks (GAT) are a class of graph neural networks that introduce an attention mechanism to dynamically weight the contributions of neighboring nodes when aggregating information. Unlike traditional Graph Convolutional Networks (GCN), which use fixed aggregation functions (e.g., mean or sum) over all neighbors, GAT assigns different importance to each neighbor through learnable attention coefficients. This enables GAT to capture more nuanced structural dependencies and focus on the most relevant nodes, making it particularly effective in tasks where relationships are heterogeneous and context-dependent.

In our work, we do not use the standard GAT implementation provided in PyG. Instead, we adopt a custom GAT inspired by QA-GNN, which explicitly incorporates edge features into the attention mechanism. The attention coefficient between nodes i and j is computed as:

$$\alpha_{i,j} = \frac{\exp\left(\mathbf{a}_s^\top \Theta_s \mathbf{x}_i + \mathbf{a}_t^\top \Theta_t \mathbf{x}_j + \mathbf{a}_e^\top \Theta_e \mathbf{e}_{i,j}\right)}{\sum_{k \in \mathcal{N}(i) \cup \{i\}} \exp\left(\mathbf{a}_s^\top \Theta_s \mathbf{x}_i + \mathbf{a}_t^\top \Theta_t \mathbf{x}_k + \mathbf{a}_e^\top \Theta_e \mathbf{e}_{i,k}\right)}.$$

This formula represents the attention score $\alpha_{i,j}$ in a GAT model that incorporates edge features. The terms $\mathbf{x}_i$ and $\mathbf{x}_j$ are the feature vectors of nodes i and j, transformed by Θ_s and Θ_t, while $\mathbf{e}_{i,j}$ represents the edge features, transformed by Θ_e. The learnable attention weights $\mathbf{a}_s, \mathbf{a}_t$, and $\mathbf{a}_e$ control the contributions of the source node, target node, and edge features. The numerator computes the unnormalized attention score, and the denominator applies the Softmax function over all neighbors, ensuring that the scores sum to 1. This mechanism allows the model to learn both node and edge importance in the graph.

This customisation-like approach also enables us to adapt the attention mechanism to task-specific requirements such as different node types, heterogeneous edge relationships, and dynamic graph structures. By modifying how node and edge features contribute to the attention score, we can fine-tune the model for different types of document structures, ensuring that our approach remains adaptable across a variety of datasets and problem Settings.

Our approach overcomes the limitations of the traditional GCN, which mainly relies on node features and assigns the same weight to all edges. Whereas in tasks like document VQA, where structural and relational information between text fragments is critical, our approach is able to capture more fine-grained contextual dependencies. This enhanced expressive power improves information aggregation and helps the model build richer representations of document layouts, logical relationships, and cross-page dependencies.

Multi-head attention is an important mechanism in attention-based models such as Transformers and graph Attention Network GAT. It works by computing multiple attention score sets in parallel. Like other multi-head attention mechanisms, we

spread the dimensions across multiple layers. The ability of the model to capture different relational aspects is enhanced. Each attention head works independently and learns different representations of the input data. This allows the model to focus on different parts of the input simultaneously. Then, the output of each head is aggregated by concatenation or averaging.

The advantage of multi-head attention is that it enhances the expressive power of the model and helps the model capture more complex and nuanced dependencies. This is particularly important in graph structure tasks where interactions between nodes and types of edges vary widely. Furthermore, multi-head attention also improves robustness to noisy inputs, as relying on multiple attention modalities reduces reliance on a single modality. With multiple attention heads, the model can better retain diverse information at each layer.

In the context of Graph Attention Networks, multi-head attention plays a vital role in handling graph-structured data. Rather than aggregating information from neighboring nodes using a single attention function, multi-head attention in GATs applies multiple independent attention mechanisms to the same graph, capturing different aspects of node interactions. This approach helps the model better understand diverse relationships in the graph. The ability to adaptively assign different attention weights to various edges further improves the model's robustness, especially when dealing with heterogeneous graphs. Moreover, multi-head attention enhances feature extraction, as each head can focus on learning different latent representations of node interactions, leading to a more expressive model.

The number of attention heads will affect the expressive power and computational cost of the model. When the number of heads is small, each attention mechanism needs to capture a wider range of relationships, which may lead to less detailed data representation, but the computational overhead is small. When the number of heads is large, the model can learn more diverse node relationships, but it also increases the computational cost and may cause redundancy if not well tuned. The key is to find the right number of heads for the specific task. We will focus on this in detail in the experiment.

Chapter 5

EXPERIMENTS and RESULTS

This chapter reports on the experiments conducted to evaluate the various components of the proposed model, as introduced in the Methodology section. The approach is based on a Graph Attention Network (GAT) that employs a dynamic scoring mechanism to capture information from adjacent nodes. The experiments systematically investigate several design choices, including the formulation of the GAT layer, the selection of the number of attention heads, the integration of the GAT layer within the internal structure of the Graph Neural Network (GNN), and the overall construction of the GNN.

This section also outlines the evaluation metrics and the tuning of hyperparameters of all method. that are critical to assessing model performance. Different configurations were tested to assess their impact on the final outcomes, and the experimental results obtained through these varied attempts are presented alongside comparisons to several published models. Details regarding the datasets used, as well as further specifics on the evaluation metrics and loss parameters, can be found in the corresponding sections.

5.1 Evaluation and Metric

This task involves multi-label prediction, meaning that the correct answer to a question may not be unique and could even be zero.

In the evaluation, following the approach of the PDF-VQA paper, the metric used is the **mean accuracy** across all questions. The accuracy of a single question (Acc) is marked as 1 only if all predictions **exactly match** the ground truth labels (0 or 1). This is formulated as:

$$\text{Acc} = \begin{cases} 1, & \text{if prediction} = \text{ground truth} \\ 0, & \text{otherwise} \end{cases}$$

The mean accuracy is then calculated as the average accuracy over all questions:

$$\text{Mean Accuracy} = \frac{1}{N} \sum_{i=1}^{N} \text{Acc}_i$$

where N is the total number of questions.

This exact match requirement presents a significant challenge, as it demands precise alignment between predictions and ground truth labels. Unlike single-label tasks, where partial correctness might be acceptable, multi-label prediction requires all predicted labels to be correct simultaneously.

This strict evaluation criterion ensures high-quality predictions but also increases the difficulty of achieving high accuracy. Furthermore, the variability in answer locations (e.g., titles, subtitles, or multiple occurrences) adds more complexity, as the model must accurately identify and extract relevant information from diverse contexts. These factors collectively make the task both challenging and meaningful for advancing document understanding and multi-label prediction capabilities.

5.2 Baseline Researches

PDF-VQA research has explored numerous baseline models to evaluate the challenges of document-based visual question answering (Task C). In addition to these baselines, various alternative approaches and models have been explored named LoSpa(Logical and Spatial Graph-based model) in Kaggle competitions, further proposing more solutions for this task.

Model	Features					Val	Test
	V.	C.	LR.	SR.	Sim		
VisualBERT	✓	×	×	×	×	21.55%	18.52%
ViLT	✓	×	×	×	×	10.21%	9.87%
LXMERT	✓	×	×	×	×	16.37%	14.41%
LoSpa	✓	✓	✓	✓	×	30.21%	28.99%
Polito's	×	✓	×	×	×	29.95%	34.52%
MemSum-DQA	×	✓	✓	×	×	40.79%	39.73%

Table 5.1: Baseline Model on the PDF-VQA dataset, Task C.

In models of PDF-VQA, 5.1 shown the answer of them, where the features include V. (Visual appearance), C. (Context), LR. (Logical Relational Information), SR. (Spatial Relational Information) and S (Similarity). All of them process the questions in the same way as the sequence of question words encoded by pre-trained BERT[4] but differ in processing other features. The three large vision-and-language pretrained models (VLPMs): VisualBERT[32], ViLT[33], and LXMERT[34] achieved better performances than other baselines with inputting only question and visual features. The result of them provide us with traditional model baseline metrics.

PDF-VQA dataset research also raised their Model: LoSpa (Logical and Spatial Graph-based model) [27]. Specifically, BERT[4] is used for text processing, while pretrained ResNet-101[9] handles images corresponding to specific page numbers. By capturing relationships between document elements via logical and spatial graphs, enhancing these relations with **GCN**, and then fuse with **visual features**. The fused representations are then utilized within a decoder-based model to explore the relationships between questions and answers. achieves the higher performance compared to all baselines, confirmed the effectiveness of their adopted **GCN-encoded relational features**. Compared to other results of Task A and Task B of the PDF-VQA reasearch, they discovered the performances on Task C are the lowest for all models. As they said in the PDF-VQA paper[27], they expressed this answers indicates the difficulty of document-level questions and produces massive room for improvement for future research on this task.

In the Kaggle competitions PDF-VQA (CIKM 2023), Polito's researchers proposed a method[35] called "Enhancing BERT-Based Visual Question Answering through Keyword-Driven Sentence Selection." This approach extracts sentences that describe images or text and uses them to pre-train a language model.

Another method, MemSum[36], works by attaching the question and type prefix to text blocks. It then applies an improved MemSum model, which combines a local encoder (LSE), a global encoder (GCE), and an extraction history encoder (EHE). The model iteratively predicts probabilities to dynamically select relevant sections or subsections until a stopping condition is met or a maximum of fmy answers is reached.

Both Polito's method and the MemSum-based DQA approach performed well, ranking second and first in the competition, respectively.

However, these models either rely on a single modality or encode text and visual features separately before combining them, without truly leveraging a cross-modal approach. The GCN in LoSpa shows the potential of graph-based integration, but its weights remain static, lacking dynamic adaptation. Additionally, LoSpa does not establish connections between images across the entire document, as the PDF-VQA dataset only provides intra-page parsing. These limitations may explain why the model's performance is still constrained.

5.3 Training Configuration

The model is optimized using AdamW [37] with carefully tuned regularization parameters to prevent overfitting in multi-label classification. Table 5.2 summarizes the key training hyperparameters.

Parameter	**Value**
Optimizer	AdamW
Base Learning Rate	5×10^{-5}
Weight Decay	1×10^{-5}
Loss Function	BCEWithLogitsLoss
Batch Size	1
Training Epochs	50
Learning Rate Scheduler	Step Decay ($\gamma = 0.5$, step=10)

Table 5.2: Training Configuration Details

Key implementation details:

- **Loss Function**: Binary cross-entropy with logits (BCEWithLogitsLoss) handles multi-label classification effectively by independently computing probabilities for each class.
- **Regularization**: Combined weight decay ($\lambda = 10^{-5}$) and gradient clipping prevent gradient explosion while maintaining stable updates.
- **Learning Rate Schedule**: Implements step decay with $\gamma = 0.5$ every 10 epochs, providing gradual refinement while maintaining training stability.
- **Batch Processing**: Batch size equal to 1 means that PyG dataloader makes every document as one graph.

All experiments were conducted on NVIDIA Tesla P100 GPU using PyTorch 2.0 with full precision training. Model checkpoints were selected based on validation accuracy with early stopping patience of 5 epochs.

5.4 Graph Design Analysis

5.4.1 Embedding Model Selection

First step of Graph design is to select an appropriate cross-modal model to represent our node features. To evaluate whether cross-modal models can effectively map images and text into a unified embedding space, I selected several visually similar images from the dataset. Some of these images originate from the same file but differ in key descriptive details. Descriptions of the images were generated using large language models and then manually refined to better reflect the image content. I tested multiple models, including CLIP[11], JINA-CLIP[12], EVA-CLIP[13], BLIP[16] extractor from lavis[31], E5-V[22], and E5-V (table:text), to extract embeddings by integrating these descriptions with the visual content.

Table 5.3: Table with color marking for matched the right desciption (Y, green) and did not matched the desciption (N, red)

Image	**Clip**	**JINA-Clip**	**EVA-Clip**	**Blip**	**E5-V**	**E5-V (table:text)**
Table1	N	Y	Y	N	N	Y
Table2	N	N	Y	N	N	Y
Table3	Y	Y	Y	Y	N	Y
Table4	N	N	N	N	Y	Y
Table5	Y	Y	N	N	Y	Y
Image1	Y	N	Y	Y	Y	Y
Image2	Y	Y	Y	N	Y	Y
Image3	Y	Y	Y	Y	Y	Y
Image4	N	N	Y	Y	Y	Y
Image5	Y	N	N	N	Y	Y

The results, as shown in the appendix table5.3, marked with color and N/Y. It indicate that E5-V[22] performs consistently well for images and chart. Notably, E5-V showed poor performance in extracting embeddings directly from table images. To address this limitation, the table content was converted into text descriptions and embedded using the E5-V (table:text) approach. Specifically, the table descriptions were embedded using a template formatted as:

Table prompt:

```
<text> "Summary above table:"
```

Text and Image prompt modified according to paper of E5-V[22]:

Text prompt:

```
<text> "Summary above text:"
```

Image prompt:

`<image> "Summary above image:"`

This modification left the information that it is a table and also more attention to the text content of the table, so that improved performance, making E5-V (table:text) the best-performing model for table-related embeddings. These results indicate that E5-V can be effectively utilized for my task requiring cross-modal understanding.

5.4.2 Similarity Graph Design

I conducted an analysis on a sample document to assess the similarity quality at different top-k similarity levels. Table 5.4 below presents the mean and minimum similarity values for various top-k selections.

k	Mean Similarity Value	Min Similarity Value
3	0.6846	0.4844
5	0.6587	0.4302
8	0.6435	0.4241
10	0.6052	0.4141

Table 5.4: Mean and minimum similarity values for different top-k selections.

Mean and minimum similarity values for different top-k selections. The table shows the mean and minimum similarity values observed for various k selections when constructing the similarity graph. As k increases, the overall similarity decreases, as expected, but the graph becomes more complex and includes more connections between nodes. This table provides a quantitative overview of how similarity values change with different top-k selections, which is crucial for understanding the trade-off between graph complexity and similarity in our approach.

We then evaluated the performance for different values of k, as shown in Table 5.5.

***k* (Top of similarity)**	**Performance (val)**
3	26.70%
5	**27.54**%
8	27.02%
10	24.44%

Table 5.5: Top-k Selection Impact on Validation Accuracy

Selecting a smaller k results in a simpler graph with higher similarity, which may help capture immediate relationships but might not provide enough structural complexity for the model to fully learn and generalize. On the other hand, a larger k leads to a more intricate graph with lower similarity, which can introduce noise and reduce the relevance of relationships.

Based on my analysis, $k = 5$ provides the optimal balance. A value below 5 simplifies the graph too much, potentially limiting the model's ability to learn useful relationships, while a value above 5 reduces the overall similarity, making the graph unnecessarily complex. This analysis highlights the need to carefully balance graph complexity and similarity for optimal model performance. Therefore, I recommend $k = 5$ as the best choice, as it strikes the right balance between capturing meaningful relationships and maintaining graph complexity.

5.4.3 Logical Relation Graph Design

As discussed in the methodology, I followed the logical relation graph from PDF-VQA, which was also used in LoSpa[27]. However, our approach is different because we extend the relationship **to document-level** diagrams. This shift allows us to capture relationships between pages and improve the model's ability to handle more complex document-wide queries.

Scope Level	**Performance (val)**
Page-level	26.33%
Doc-level	27.19%

Table 5.6: Results of Logical Relation Graph at Page-level and Doc-level.

In our tests, we noticed an improvement in performance, especially when dealing with cross-page question-and-answer tasks. By extending the logical diagram to the document level, we are able to better understand the relationships between objects across different pages. This adjustment enhances the model's ability to handle cross-page problems, which is especially important for tasks that require a complete understanding of the document.

While the performance boost is not particularly dramatic, it demonstrates the value of considering document-level relationships. This change allows us to take full advantage of the context of the entire document, which improves accuracy when dealing with complex problems involving information from different parts of the document. The results show that document-level logical relationships have a positive effect on performance, especially in tasks requiring cross-page reasoning, which makes this method of great significance in improving the performance of cross-page question answering.

5.4.4 Spatial Relation Graph Design

In Table 5.7, we present the results of the spatial relation graph at both the page-level and document-level for different neighbor numbers.

Neighbor number	Scope Level	
	Page-level	**Doc-level**
2	18.23%	18.98%
4	17.56%	17.72%

Table 5.7: Results of Spatial Relation Graph.

From the table, we can observe that the performance at the document-level is slightly better than at the page-level. Specifically, with 2 neighbors, the document-level graph achieved 18.98% accuracy compared to 18.23% at the page-level. Similarly, with 4 neighbors, the improvement from the page-level (17.56%) to the document-level (17.72%) is still modest. These results suggest that the document-level spatial relations bring a small yet positive benefit over page-level relations.

However, it is important to note that the improvement in performance is relatively limited. This might be attributed to the fact that our model heavily relies on the quality of the graph structure itself, especially since we do not have an additional decoder component, as seen in other models such as Lospa or Doc-GCN. These models use decoders to further learn the relationships in the question-answering task (QA), allowing them to refine and adjust the graph dynamically for more effective reasoning. Without such a decoder, our model lacks the ability to explicitly learn and adapt to the underlying QA relationships, which could explain why the document-level graph, while beneficial, does not yield a larger improvement.

In models like Lospa and Doc-GCN, the decoder enables the model to iteratively refine the graph's utility in relation to the task at hand (i.e., answering questions). This additional layer of learning provides a more robust representation of the document and its spatial relations. Since our model does not possess a decoder to refine the graph for the QA task, the benefits of extending spatial relations from page-level to document-level may not be fully realized. Therefore, the improvements we see are likely constrained by the limited ability to adapt and optimize the graph structure specifically for question-answering purposes.

5.5 Model Architecture Analysis

At beginning of choosing the suitable construct, other parameters are set as:

Parameter	Value
Neighbor for SR Graph	2
Top-k in Similarity Graph	5
Number of GAT layer in GNN	1
Number of GAT heads	1

Table 5.8: Parameter Initialization

The table above presents the initial parameter settings used in our experiments. The first two parameters, "Neighbor for SR" and "K of the top of similarity," were determined based on the analysis of the graph design. Specifically, the optimal number of neighbors for the spatial relation (SR) was set to 2, and the top-k similarity value was chosen to be 5. These settings were selected after evaluating the performance trade-offs between graph simplicity and the strength of semantic connections.

The remaining two parameters, "GAT layer num of GNN" and "multi head of GAT," are part of the model analysis and will be explored further in our experiments. The number of GAT layers is initially set to 1, and the multi-head attention mechanism is also set to 1. These choices will be tested and refined in the model analysis phase to evaluate their impact on performance and improve the model's ability to capture complex relational patterns in the graph.

5.5.1 GNN Composition

Results are below where LR and SR are the Structural relations from document, LR means my doc-level logical Parent-child Relation information, and SR means my doc-level Spatial Relation information.

From the experimental results of Tab.5.9, initially, I observe that individual graphs, with accuracy ranging from 18.23% on neighbor of SR graph, and around 27% on similarity and logical relation of parents and children, confirming their ability to capture information on different graph. Similarity and LR graph is more meaningful for individually learning.

The global approach, which incorporates both Logical Relation (LR) and similarity graphs, achieves the highest accuracy of 30.45%. This result underscores the importance of combining different types of graph structures, where the LR graph captures hierarchical and structural relationships, and the similarity graph captures content-based affinities. These graphs provide complementary semantic

Method	Performance (val)	
	Composition	**Mean Acc**
Individual Graph Learning	LR Graph	27.19%
	SR Graph	18.98%
	Similarity Graph	27.54%
Global Graph Learning	LR & SR Graphs	26.33%
	LR & Sim Graphs	**30.45%**
	SR & Sim Graphs	26.85%
	3 Graphs	25.91%
Combined Graph Learning	LR & SR Graphs (2GNNs)	20.14%
	LR & Sim Graphs (2GNNs)	25.65%
	SR & Sim Graphs (2GNNs)	21.86%
	3 Graphs (3GNNs)	26.68%

Table 5.9: Results with Different Graph Constructions

perspectives that enhance the model's ability to understand and process document-level information. The single-GNN setup effectively integrates these two graphs, striking an optimal balance between information richness and simplicity without introducing unnecessary complexity.

This compares to the performance degradation observed in three-graph combinations (25.91%) and multi-GNN architectures (from 20.14% to 26.68%), highlighting the challenges posed by increasing model complexity. While more graphs or GNN layers might seem like they should increase the learnability of the model and thus improve performance, they can amplify the noise and make the results worse. This can be attributed to the inherent modular nature of GAT, which makes it difficult to find an optimal way to combine the results of multiple components. Not all graph combinations are cooperative; Some combinations may introduce conflicting signals that can confuse the model. Therefore, we conclude that multi-GNN architectures require careful design of coordination mechanisms, and achieving better results may require further ensuring feature alignment between different layers and graphs.

These findings highlight that successful graph fusion relies not just on stacking multiple components but on ensuring that the graphs are informationally complementary and that the overall architecture remains simple and effective. The synergy between the LR and similarity graphs likely leads to superior performance because it combines structural dependencies with content similarities in a unified way. This approach avoids the risks of overfitting and noise interference that can arise from overly complex models or poorly coordinated multi-GNN systems.

5.5.2 GNN Structure

We have discussed what kind of construction to use for different graph combinations and have found a suitable one. Now, we take a step further to refine the internal design of the GNN.

GNN Composition	Performance (val)	
	GAT Layer	Mean Acc
LR & Sim Graphs Global GNN	1	30.45%
	2	16.18%
	3	13.77%

Table 5.10: Performance of global GNN with LR and similarity graphs

The number of graph attention (GAT) layers significantly impacts information propagation in my global LR-similarity GNN. While deeper networks theoretically enable multi-hop reasoning, my experiments reveal diminishing returns with additional layers (**1-layer: 30.45% vs 3-layer: 13.77%**). This suggests two fundamental constraints:

First, the shallow semantic hierarchy in my graphs differs fundamentally from QAGNN's knowledge graph requirements. Where QAGNN needs deep layers to aggregate distant concepts (e.g., inferring "Einstein" → "relativity" → "nuclear energy"). In my graph similarity measurements creates symmetric feature, indicate the shallow networks sufficiently capture.

Second, the direct pairwise relations in my graph structure lack hierarchical dependencies requiring deep propagation. Although multi-layer GNNs benefit tasks needing complex inference chains (e.g., multi-step logical reasoning), my task primarily relies on immediate neighbor interactions rather than long-range dependencies.

The unsatisfactory performance may be due to two mechanisms: (1) *Over-smoothing*, that is, repeated information transmission makes the node features converge, ignoring the distinguishing details; (2) *Noise amplification*, that is, unrelated side connections are accumulated through multiple propagation steps. These findings highlight the importance of matching GNN depth to graph connectivity patterns and task requirements.

Based on this conclusion, suggests that adding more layers did not lead to further improvements. I kept the number of GAT layers at 1, limiting graph learning to a single step. The best result remained at 30.45%.

5.5.3 GAT Analysis

After finalizing the GNN structure, we further explore the optimal multi-head configuration for the Graph Attention Network (GAT). In transformer-based models (e.g., BERT, GPT), multi-head attention typically uses 8–16 heads, balancing model capacity and computational efficiency. For GAT, the choice of heads depends on graph complexity, feature diversity, for my task and also graph complexity.

Analysis of Multi-Head Performance:

my experiments yield the following validation performance across head configurations:

GNN Composition	Performance (val)	
	Number of Heads	Mean Acc
LR & Sim Graphs Global GNN	1	30.45%
	4	**31.50%**
	8	30.46%
	16	30.29%

Table 5.11: Performance of global GNN with LR and similarity graphs

The best performance with 4 heads can be attributed to a balance between computational efficiency and representation diversity. While a single head lacks the capacity to capture diverse attention patterns, increasing the number of heads beyond 4 introduces redundancy and noise, particularly in sparse graph structures. Specifically, 4 heads provide sufficient parallel attention mechanisms to model complex relationships in my `LR&sim` graphs without overfitting or oversplitting semantic contexts. In contrast, higher head counts (e.g., 16) amplify irrelevant features and reduce discriminative power, leading to performance degradation.

The relatively **small differences** in performance across head configurations can be explained by the nature of my task and dataset. First, the graph structure in my problem is not highly complex, meaning that even a single head can capture a significant portion of the relevant relationships. Second, the attention mechanism in GAT inherently aggregates information from neighboring nodes, which reduces the sensitivity to the exact number of heads. Finally, the task itself may not require extremely fine-grained attention patterns, making the model less dependent on the specific choice of head count. This suggests that while tuning the number of heads is important, the overall architecture and graph properties play a more dominant role in determining performance.

5.6 Result and Comparison

After analyzing different models and parameter settings, we identified the optimal approach that achieves the best performance in my task with the parameter shown in Tab. 5.12.

Parameter	**Value**
Neighbor for SR Graph	2
Top-k in Similarity Graph	5
GNN Model Composition	LR and Similarity Graphs with Global GNN
Number of GAT layer in GNN	1
Number of GAT Heads	4

Table 5.12: Configuration Model and Graph

Table 5.13 presents a comparison of different methods on the PDF-VQA dataset, specifically for Task C. The table evaluates various models based on their incorporated features and their performance on both the validation and test sets. The features considered include V (Visual appearance), C (Contextual text information), LR (logical Parent-child Relation information), SR (Spatial Relation information) and Sim (Similarity information).

Model	Features					Val	Test
	V.	C.	LR.	SR.	Sim		
VisualBERT	✓	×	×	×	×	21.55%	18.52%
ViLT	✓	×	×	×	×	10.21%	9.87%
LXMERT	✓	×	×	×	×	16.37%	14.41%
LoSpa	✓	✓	✓	✓	×	30.21%	28.99%
Polito's	×	✓	×	×	×	29.95%	34.52%
MemSum-DQA	×	✓	✓	×	×	40.79%	39.73%
Mine	✓	✓	✓ (doc-level)	×	✓ (doc-level)	**31.50%**	**38.72%**

Table 5.13: Comparison on the PDF-VQA dataset, Task C.

I proposed method integrates all five features (V, C, LR and Sim) within a global GNN framework. By jointly leveraging visual, textual, Logical relational, and similarity-based signals, the model achieves the valuable results: 31.50% on validation set and 38.72% on test set.

The results of VisualBERT[32], ViLT[33], and LXMERT[34] indicate that whether using segmented document elements for object-level visual feature extraction, image patch-based representations, or incorporating bounding box features,

these approaches only focus on understanding the image modality. As a result, their ability to answer questions about the document remains very limited

As seen in the LoSpa which constructs logical and spatial graphs to enhance relational information between document elements using Graph Convolution Networks (GCN). The model combines BERT and ResNet-101 to extract textual and visual features, and employs Transformer encoder-decoder with a pointer network for QA prediction. Their results (30.21% / 28.99%), this approach provides a reasonable starting point of graph learning but remains limited in fully capturing complex interrelations between modalities.

My function solved the issue that visual and text information are separated processed. I introduced cross-modal learning, allowing different types of data to be mapped into a shared semantic space, where both relational and similarity-based information can be jointly used for learning, that is did not ever used in baseline models.

The result of LoSpa using GCN shows the limitation of aggregation of the adjacent node in the graph. To improve aggregation, I added Graph Attention Networks (GATs), allowing the model to learn dynamic adjacency weights and assign different levels of importance to neighboring nodes. The result indicated that this design allows the model to learn not only explicit relational structures but also to identify key parent-child relationships and high-similarity adjacent nodes.

More importantly, my approach removes traditional page-wise segmentation constraints. Even when relevant text passages and images are located on different pages or distant sections, their semantic connections can be effectively captured and modeled. By combining explicit relational modeling with implicit cross-modal similarity modeling, my model greatly improves joint understanding of text and images, as well as semantic and structural comprehension of documents, leading to better question-answering accuracy. Solved the mainly problem of Task C in pdf-vqa[27].

Although Polito's method, pretraining on golden sentences, and MemSum method have shown significant effectiveness in processing document text. AT the same time, my results did not surpass MemSum. However, my model performed better on a larger test set, indicating that it is more data-driven and has stronger generalization capabilities. Moreover, my model can be adapted to more learning tasks, enabling it to handle more complex visual question-answering problems. There is also significant space for improvement. If in the future with the development of more advanced parsing models, more complex logical relationships, or additional useful graph-based information, the model's learning ability can be further enhanced.

Contribution

After obtaining the answers and comparing them with other models, we can conclude that my cross-modal learning approach fundamentally reshapes the model's information integration mechanism:

- **Cross-modal Alignment**: By mapping data into a shared semantic space, the model no longer processes text or images in isolation, enabling multi-source information to complement and validate each other;
- **Innovation in Document-Level Graph Modeling**: While conventional methods are limited to local intra-page relationships, my graph structure spans the entire document. Leveraging both semantic similarity (e.g., term repetition) and structural relevance (e.g., parent-child paragraphs) to drive reasoning;
- **Dynamic Weight Learning**: GAT's adaptive attention mechanism enables the model to distinguish critical nodes from redundant content without relying on manually defined rules, significantly enhancing flexibility in complex document scenarios.

The performance improvements (31.50% validation accuracy, 41.01% test score) validate the effectiveness of joint semantic-structural modeling: The model not only captures explicit relationships (e.g., "Figure 3" and its description paragraph) but also builds globally consistent understanding through implicit similarities (e.g., topic-related terms distributed across multiple pages)—a breakthrough unattainable by traditional single-modality or page-level approaches.

Chapter 6

CONCLUSIONS and FUTURE WORK

We have now reached the final stage of this research. In this chapter, we will provide a comprehensive summary based on our experimental results. Our model has achieved a certain level of improvement, although it still falls short of surpassing the current state-of-the-art model. This not only demonstrates the feasibility of the Structural-Semantic Dynamic Graph Learning method for Document Visual Question Answering but also highlights the existing limitations that need to be addressed.

However, the true strength of our model lies in its scalability and future potential. Given that the approach is well-founded, we can continue refining the graph structure and optimizing our strategies to further enhance performance. In this chapter, we will also discuss some of the shortcomings in our work, as well as potential areas for improvement.

For future development, we hope that our findings will serve as a valuable reference for subsequent research and contribute to advancements in the field. By continuously improving our approach, we believe it is possible to bridge the gap between textual and visual information more effectively, paving the way for more advanced multimodal AI systems.

6.1 Conclusion

Our approach begins by extending data processing from page-level to document-level, and experimental results demonstrate that this significantly enhances task performance. Unlike methods that rely solely on individual page information, our approach integrates content across the entire document, providing a more comprehensive context. This improvement not only strengthens the model's ability to understand the logical structure of a document but also enhances the connections between different pieces of information, leading to greater interpretability. By capturing interactions among elements throughout the document, our method enables more effective reasoning over a broader scope, ultimately boosting overall performance.

In addition to document-level processing, our model leverages cross-modal embeddings, which allow visual information to be more effectively utilized. This approach enhances the model's comprehension of document content by integrating textual and visual modalities in a more cohesive manner. While we did not construct single-modal embeddings or develop a separate graph for direct comparison, our results clearly demonstrate the value of cross-modal learning. Although testing single-modal embeddings under the same conditions remains an open question, our work underscores the significant performance gains enabled by cross-modal embeddings, offering new possibilities for integrating multimodal information in future research.

To further refine information aggregation within document structures, we incorporated Graph Attention Networks (GAT), enabling dynamic weight adjustments when modeling relationships between different elements. We also explored multiple GAT attention score computation methods and evaluated their impact across different task settings. By leveraging GAT, we established a more flexible framework that ensures key information is effectively propagated through the graph. This adaptive weighting mechanism improves the model's learning capacity, making it better suited to capturing critical content in complex documents.

A comprehensive evaluation of different model architectures was conducted, examining the role of various graph-based components in cross-modal tasks. Our systematic assessment of different graph neural network structures revealed notable improvements in performance. Furthermore, comparative experiments against existing baselines confirmed the effectiveness of our approach—our model outperformed the PDF-VQA baseline by nearly 10% on the test set, highlighting the potential of combining cross-modal embeddings with graph neural networks for document question answering tasks.

Beyond these immediate improvements, our research opens up broader avenues for advancing multimodal learning. By effectively integrating cross-modal representations with graph neural networks, our findings provide a solid foundation for

enhancing document question answering, information retrieval, and other multimodal applications. A key direction for future work involves scaling our approach for real-world deployment, optimizing the model for real-time processing and increased efficiency in large-scale production environments.

Another promising avenue involves integrating our method with retrieval-augmented generation (RAG) systems and advanced generative large models (such as GPT-4, Gemini, and Claude). Our Structural-Semantic Dynamic Graph Learning for Document Visual Question Answering can serve as a retrieval mechanism, improving structured document search and content retrieval for RAG frameworks. In addition, as generative models are increasingly used in inference tasks, our methods can enhance their ability to analyze document structure, align text with visual elements, and support more precise query interpretation. These contributions point to an exciting future in which multimodal retrieval and structured reasoning will play a greater role in AI-driven document understanding.

Looking further ahead, the cross-modal representation learning paradigm introduced in our work has potential applications beyond document question answering. By refining multimodal information retrieval, this approach could support tasks such as legal document analysis, financial report synthesis, and large-scale academic knowledge extraction, where seamless alignment between text and visual components is essential. Furthermore, our model can be incorporated into intelligent document processing workflows, automating processes like document classification, content indexing, and interactive knowledge retrieval, thereby benefiting industries such as corporate governance, digital archiving, medical records analysis, and regulatory compliance.

While our research has demonstrated strong empirical results, there remain many opportunities for further advancements. Future work could focus on improving model generalization across diverse document types, reducing computational overhead for real-time applications, and integrating our approach into large-scale enterprise document processing pipelines. By continuing to refine and expand upon this research, we contribute to the broader goal of developing more intelligent, interpretable, and scalable cross-modal AI systems for document understanding.

6.2 Future work

Future research presents numerous promising directions, particularly given the strong generalization capability of our graph learning approach, which allows it to adapt well to different tasks and data conditions. This generalization ability stems not only from the intrinsic properties of Graph Neural Networks (GNNs) but also from the inherent flexibility of graph structures. Graphs are highly adaptable representations that can model various entities and their relationships using nodes and edges, making them suitable for diverse data modalities and application scenarios. Because of this flexibility, we can define different types of graphs based on specific task requirements, such as document-level, paragraph-level, or even cross-document graph structures. This adaptability enables GNNs to maintain strong performance across different learning environments. Additionally, graph learning integrates structural, semantic, and cross-modal information, enhancing the overall expressiveness of representations and providing a unified framework for cross-domain tasks. This not only improves the model's ability to capture complex interactions but also strengthens its capability to generalize to new tasks or unseen data. The combination of flexibility and generalization makes graph-based learning an essential approach in multimodal information processing, document analysis, and knowledge modeling.

In future work, It can be further explore for various graph learning techniques to enhance the model's understanding of document structures. Currently, our approach connects page-level information at the document level; however, if we can further optimize OCR extraction or improve the completeness of information extraction—for example, by adopting a document-level training approach similar to PDF-VQA using Mask R-CNN—the model's grasp of overall document structure could be significantly enhanced. Additionally, our current spatial graph modeling did not achieve the desired performance, likely due to insufficient utilization of spatial information. Incorporating richer positional data—such as by constructing adaptive edge weights based on relative positions or leveraging Transformer-based relative position encoding—could allow the model to capture the layout relationships between elements within a document more accurately, thereby improving the alignment of information across pages.

Furthermore, furture research can deepen our graph representations and enrich the semantic modeling of edge features. At present, edge information is primarily based on structural and spatial relationships; in the future, we could define edge semantics more explicitly through natural language descriptions. Techniques like text-based relation extraction could be used to build more expressive edge features that are then embedded into the GNN learning process. Our model benefits from the dynamic aggregation capabilities of Graph Attention Networks (GAT), which allow adaptive adjustment of node and edge weights during training. Exploiting

this feature to more closely integrate cross-modal information—such as textual and visual features—into the dynamic adjustment of edge weights could further improve the model's ability to capture inter-element relationships and enhance overall reasoning performance.

On the experimental side, future work can conduct more comprehensive ablation studies and parameter tuning to systematically investigate the contribution of each module and optimize the overall architecture. In particular, a deeper examination of various attention score computation methods within GAT—such as dot-product attention or cosine similarity-based attention—will be conducted to determine the most suitable mechanism for document question answering tasks. This approach is expected to not only improve GAT's feature aggregation capabilities in complex document structures but also enhance the integration of cross-modal information within the GNN, ultimately boosting the system's stability and scalability.

Moreover, additional avenues for future work can be pursued from multiple perspectives. We intend to investigate the scalability of our proposed method in large-scale, real-world scenarios, as well as assess its robustness under noisy or low-quality input conditions. Collaborations with industry partners to integrate this technology into commercial systems are also planned, along with the public release of our code and datasets to foster further research and collaboration across academia and industry. Continuous improvements based on community feedback and emerging technologies will guide the evolution of this method, ensuring its competitiveness across various multimodal tasks. We firmly believe that our work lays a solid foundation for advancing multimodal document analysis and provides invaluable technical and theoretical insights for the development of more robust, efficient, and interpretable AI systems in the future.

Going forward, sustainable and impactful applications of this research are aligned with global goals such as the Sustainable Development Goals (SDGs). In terms of quality education (SDG 4), promoting document understanding and knowledge retrieval through intelligent models can enable the popularization of structured learning materials, facilitate the digitization of educational resources and personalize learning experiences. In industry, Innovation and Infrastructure (SDG 9), the ability to efficiently analyze and process multimodal documents can drive innovation in business intelligence, legal document processing and digital governance. In addition, in the context of Peace, Justice, and Strong Institutions (SDG 16), improved document understanding models contribute to transparent information retrieval, fraud detection, and legal text analysis, thereby enhancing institutional integrity.

By continuously refining this research direction, future AI systems can become more interpretable, scalable, and applicable across multiple domains. Collaboration between academia and industry will be crucial in integrating these developments into real-world applications, fostering a future where intelligent document analysis

contributes to more efficient knowledge systems, equitable access to information, and enhanced decision-making processes.

Bibliography

[1] Robin M. Schmidt. «Recurrent Neural Networks (RNNs): A gentle Introduction and Overview». In: *CoRR* abs/1912.05911 (2019). arXiv: 1912.05911. URL: http://arxiv.org/abs/1912.05911 (cit. on p. 5).

[2] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin. «Attention Is All You Need». In: *CoRR* abs/1706.03762 (2017). arXiv: 1706.03762. URL: http://arxiv.org/abs/1706.03762 (cit. on pp. 5, 8, 12).

[3] Ralf C. Staudemeyer and Eric Rothstein Morris. «Understanding LSTM - a tutorial into Long Short-Term Memory Recurrent Neural Networks». In: *CoRR* abs/1909.09586 (2019). arXiv: 1909.09586. URL: http://arxiv.org/abs/1909.09586 (cit. on p. 5).

[4] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. «BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding». In: *CoRR* abs/1810.04805 (2018). arXiv: 1810.04805. URL: http://arxiv.org/abs/1810.04805 (cit. on pp. 6, 7, 47).

[5] Yinhan Liu et al. *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. 2019. arXiv: 1907.11692 [cs.CL]. URL: https://arxiv.org/abs/1907.11692 (cit. on p. 6).

[6] Tom B. Brown et al. *Language Models are Few-Shot Learners*. 2020. arXiv: 2005.14165 [cs.CL]. URL: https://arxiv.org/abs/2005.14165 (cit. on p. 7).

[7] Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2023. arXiv: 1910.10683 [cs.LG]. URL: https://arxiv.org/abs/1910.10683 (cit. on p. 9).

[8] Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*. 2019. arXiv: 1910.13461 [cs.CL]. URL: https://arxiv.org/abs/1910.13461 (cit. on p. 9).

[9] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. «Deep Residual Learning for Image Recognition». In: *CoRR* abs/1512.03385 (2015). arXiv: 1512.03385. URL: http://arxiv.org/abs/1512.03385 (cit. on pp. 10, 47).

[10] Alexey Dosovitskiy et al. «An Image is Worth 16x16 Words: Transformers for Image Recognition at Scale». In: *CoRR* abs/2010.11929 (2020). arXiv: 2010.11929. URL: https://arxiv.org/abs/2010.11929 (cit. on p. 10).

[11] Alec Radford et al. «Learning Transferable Visual Models From Natural Language Supervision». In: *CoRR* abs/2103.00020 (2021). arXiv: 2103.00020. URL: https://arxiv.org/abs/2103.00020 (cit. on pp. 11, 14, 15, 34, 50).

[12] Andreas Koukounas et al. *Jina CLIP: Your CLIP Model Is Also Your Text Retriever*. 2024. arXiv: 2405.20204 [cs.CL]. URL: https://arxiv.org/abs/2405.20204 (cit. on pp. 13, 34, 50).

[13] Quan Sun, Yuxin Fang, Ledell Wu, Xinlong Wang, and Yue Cao. «EVA-CLIP: Improved Training Techniques for CLIP at Scale». In: (2023). arXiv: 2303.15389 [cs.CV]. URL: https://arxiv.org/abs/2303.15389 (cit. on pp. 13, 34, 50).

[14] Yuxin Fang, Wen Wang, Binhui Xie, Quan Sun, Ledell Wu, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. *EVA: Exploring the Limits of Masked Visual Representation Learning at Scale*. 2022. arXiv: 2211.07636 [cs.CV]. URL: https://arxiv.org/abs/2211.07636 (cit. on p. 13).

[15] Yuxin Fang, Quan Sun, Xinggang Wang, Tiejun Huang, Xinlong Wang, and Yue Cao. «EVA-02: A visual representation for neon genesis». In: *Image and Vision Computing* 149 (Sept. 2024), p. 105171. ISSN: 0262-8856. DOI: 10.1016/j.imavis.2024.105171. URL: http://dx.doi.org/10.1016/j.imavis.2024.105171 (cit. on p. 13).

[16] Junnan Li, Dongxu Li, Caiming Xiong, and Steven C. H. Hoi. «BLIP: Bootstrapping Language-Image Pre-training for Unified Vision-Language Understanding and Generation». In: *CoRR* abs/2201.12086 (2022). arXiv: 2201.12086. URL: https://arxiv.org/abs/2201.12086 (cit. on pp. 13, 34, 50).

[17] Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. *BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models*. 2023. arXiv: 2301.12597 [cs.CV]. URL: https://arxiv.org/abs/2301.12597 (cit. on p. 14).

[18] Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. *Visual Instruction Tuning*. 2023. arXiv: 2304.08485 [cs.CV]. URL: https://arxiv.org/abs/2304.08485 (cit. on pp. 14, 15).

[19] Soravit Changpinyo, Piyush Sharma, Nan Ding, and Radu Soricut. *Conceptual 12M: Pushing Web-Scale Image-Text Pre-Training To Recognize Long-Tail Visual Concepts*. 2021. arXiv: 2102.08981 [cs.CV]. URL: https://arxiv.org/abs/2102.08981 (cit. on p. 15).

[20] Pan Lu, Swaroop Mishra, Tony Xia, Liang Qiu, Kai-Wei Chang, Song-Chun Zhu, Oyvind Tafjord, Peter Clark, and Ashwin Kalyan. *Learn to Explain: Multimodal Reasoning via Thought Chains for Science Question Answering*. 2022. arXiv: 2209.09513 [cs.CL]. URL: https://arxiv.org/abs/2209.09513 (cit. on p. 15).

[21] Ting Jiang, Shaohan Huang, Zhongzhi Luan, Deqing Wang, and Fuzhen Zhuang. *Scaling Sentence Embeddings with Large Language Models*. 2023. arXiv: 2307.16645 [cs.CL]. URL: https://arxiv.org/abs/2307.16645 (cit. on p. 15).

[22] Ting Jiang, Minghui Song, Zihan Zhang, Haizhen Huang, Weiwei Deng, Feng Sun, Qi Zhang, Deqing Wang, and Fuzhen Zhuang. *E5-V: Universal Embeddings with Multimodal Large Language Models*. 2024. arXiv: 2407.12580 [cs.CL]. URL: https://arxiv.org/abs/2407.12580 (cit. on pp. 15, 34, 50).

[23] Jie Zhou, Ganqu Cui, Zhengyan Zhang, Cheng Yang, Zhiyuan Liu, and Maosong Sun. «Graph Neural Networks: A Review of Methods and Applications». In: *CoRR* abs/1812.08434 (2018). arXiv: 1812.08434. URL: http://arxiv.org/abs/1812.08434 (cit. on p. 16).

[24] Keiron O'Shea and Ryan Nash. *An Introduction to Convolutional Neural Networks*. 2015. arXiv: 1511.08458 [cs.NE]. URL: https://arxiv.org/abs/1511.08458 (cit. on p. 16).

[25] Thomas N. Kipf and Max Welling. *Semi-Supervised Classification with Graph Convolutional Networks*. 2017. arXiv: 1609.02907 [cs.LG]. URL: https://arxiv.org/abs/1609.02907 (cit. on p. 17).

[26] Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Liò, and Yoshua Bengio. *Graph Attention Networks*. 2018. arXiv: 1710.10903 [stat.ML]. URL: https://arxiv.org/abs/1710.10903 (cit. on pp. 18, 31, 32).

[27] Yihao Ding, Siwen Luo, Hyunsuk Chung, and Soyeon Caren Han. «PDFVQA: A New Dataset for Real-World VQA on PDF Documents». In: (2023). arXiv: 2304.06447 [cs.CV]. URL: https://arxiv.org/abs/2304.06447 (cit. on pp. 20, 21, 25, 31, 37, 47, 52, 59).

[28] Siwen Luo, Yihao Ding, Siqu Long, Josiah Poon, and Soyeon Caren Han. *Doc-GCN: Heterogeneous Graph Convolutional Networks for Document Layout Analysis*. 2022. arXiv: 2208.10970 [cs.CV]. URL: https://arxiv.org/abs/2208.10970 (cit. on pp. 31, 37).

[29] Michihiro Yasunaga, Hongyu Ren, Antoine Bosselut, Percy Liang, and Jure Leskovec. *QA-GNN: Reasoning with Language Models and Knowledge Graphs for Question Answering*. 2022. arXiv: 2104.06378 [cs.CL]. URL: https://arxiv.org/abs/2104.06378 (cit. on p. 31).

[30] Zhenrong Zhang, Jiefeng Ma, Jun Du, Licheng Wang, and Jianshu Zhang. «Multimodal Pre-Training Based on Graph Attention Network for Document Understanding». In: *IEEE Transactions on Multimedia* 25 (2023), pp. 6743–6755. DOI: 10.1109/TMM.2022.3214102 (cit. on p. 31).

[31] Dongxu Li, Junnan Li, Hung Le, Guangsen Wang, Silvio Savarese, and Steven C. H. Hoi. *LAVIS: A Library for Language-Vision Intelligence*. 2022. arXiv: 2209.09019 [cs.CV]. URL: https://arxiv.org/abs/2209.09019 (cit. on pp. 34, 50).

[32] Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. *VisualBERT: A Simple and Performant Baseline for Vision and Language*. 2019. arXiv: 1908.03557 [cs.CV]. URL: https://arxiv.org/abs/1908.03557 (cit. on pp. 47, 58).

[33] Wonjae Kim, Bokyung Son, and Ildoo Kim. *ViLT: Vision-and-Language Transformer Without Convolution or Region Supervision*. 2021. arXiv: 2102.03334 [stat.ML]. URL: https://arxiv.org/abs/2102.03334 (cit. on pp. 47, 58).

[34] Hao Tan and Mohit Bansal. *LXMERT: Learning Cross-Modality Encoder Representations from Transformers*. 2019. arXiv: 1908.07490 [cs.CL]. URL: https://arxiv.org/abs/1908.07490 (cit. on pp. 47, 58).

[35] Davide Napolitano, Lorenzo Vaiani, and Luca Cagliero. «Enhancing BERT-Based Visual Question Answering through Keyword-Driven Sentence Selection». In: (2023). arXiv: 2310.09432 [cs.CL]. URL: https://arxiv.org/abs/2310.09432 (cit. on p. 48).

[36] Nianlong Gu, Yingqiang Gao, and Richard H. R. Hahnloser. *MemSum-DQA: Adapting An Efficient Long Document Extractive Summarizer for Document Question Answering*. 2023. arXiv: 2310.06436 [cs.CL]. URL: https://arxiv.org/abs/2310.06436 (cit. on p. 48).

[37] Ilya Loshchilov and Frank Hutter. «Fixing Weight Decay Regularization in Adam». In: *CoRR* abs/1711.05101 (2017). arXiv: 1711.05101. URL: http://arxiv.org/abs/1711.05101 (cit. on p. 49).