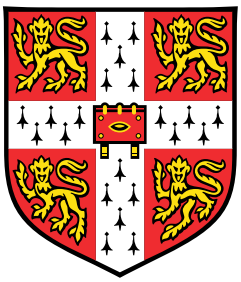
Reranking Multi-Modal Retrievers for Knowledge-Based Visual Question Answering



Felix Zhu

Department of Engineering
University of Cambridge

This dissertation is submitted for the degree of
MPhil in Machine Learning and Machine Intelligence

Wolfson College

August 2024

Declaration

I, Felix Zhu of Wolfson College, being a candidate for the MPhil in Machine Learning and Machine Intelligence, hereby declare that this report and the work described in it are my own work, unaided except as may be specified below, and that the report does not contain material that has already been used to any substantial extent for a comparable purpose.

The code for this thesis is available here. The retrieval model is implemented and trained using PreFLMR (Lin et al., 2024b)[1], with checkpoints downloaded from HuggingFace[2]. Data is sourced from the M2KR dataset created by Lin et al. (2024b) and was downloaded from HuggingFace[3]. We extended the experimentation pipeline based on RunwayExperiments[4] and, by extension, PyTorch Lightning. Additional code for defining, training and evaluating models are written in PyTorch and entirely original unless otherwise specified. All models were trained and evaluated on the Cambridge University HPC using a single A100 GPU in under 1,000 GPU hours.

Word Count: 14,933

Felix Zhu
August 2024

[1] https://github.com/LinWeizheDragon/Retrieval-Augmented-Visual-Question-Answering
[2] https://huggingface.co/LinWeizheDragon
[3] https://huggingface.co/datasets/BByrneLab/multi_task_multi_modal_knowledge_retrieval_benchmark_M2KR
[4] https://github.com/EriChen0615/runway_for_ml

Acknowledgements

I would like to express my deepest gratitude to my supervisors, Professor Bill Byrne and Weizhe Lin, for their invaluable guidance throughout my research journey. To Professor Bill Byrne, I extend my sincere thanks for his calm advice and unwavering supportiveness. His level-headed approach provided me with the stability and confidence needed to navigate the challenges of my research. I am equally grateful to Weizhe Lin for his crucial support in the early stages of my work. His assistance in refining my research direction and his constant availability to discuss results and code were instrumental in shaping this thesis. My appreciation also goes out to my fellow students at Wolfson College and in the Machine Learning and Machine Intelligence program who have been a source of inspiration and joy throughout this journey.

Abstract

Knowledge-Based Visual Question Answering (KB-VQA) integrates information from external sources by retrieving relevant documents to generate answers to questions about images. The accuracy of KB-VQA heavily relies on retrieving documents relevant to the context of the question. This thesis investigates reranking as a technique to improve retrieval recall by refining an initial list of retrieved documents. While reranking has been extensively researched in information and text retrieval, its application to KB-VQA remains unexplored.

Our primary contribution is introducing multimodal rerankers in a multi-stage retrieval pipeline for KB-VQA, demonstrating statistically significant improvements in recall over state-of-the-art PreFLMR retrievers on both OKVQA and EVQA. Our most powerful reranker, monoBLIP-2, improves the Recall@5 of PreFLMR for EVQA from 32.7% to 39.9% and for OKVQA from 27.5% to 47.5%.

We also develop a modularized reranker, ModPreFLMR, that leverages late-interaction embeddings from retrieval for efficiency. We introduce this as a concept of mid-interaction, striking a balance in the performance-efficiency trade-off between the late-interaction mechanisms of PreFLMR and the early-interaction mechanisms of monoBLIP-2. It is 97x faster than monoBLIP-2 and boosts the Recall@5 of the smallest PreFLMR-B to surpass PreFLMR-G with minimal computational overhead.

We provide a comprehensive analysis of the mechanics behind training rerankers, including various model configurations, sampling methods, and loss functions. Our findings show that rerankers benefit significantly from strong pre-training but are susceptible to overfitting. We also find significant performance gains by training on retrieved documents that represent the distribution of top retrieval candidates, although sampling a balanced ratio of positive and negative documents becomes necessary to prevent class imbalance. Additionally, we describe the theoretical and practical advantages of using a listwise loss function to rank small datasets with homogeneous documents like OKVQA and a pointwise loss function to rank larger datasets with more diverse batches like EVQA. We conclude that the best rerankers leverage strong pre-trained models, initially aligned to the reranking task with a pointwise loss on a wide range of datasets, and then fine-tuned on specific retrievers with a listwise loss.

Table of Contents

List of Figures xiii

List of Tables xvii

1 Introduction 1
- 1.1 Contributions . . . 2
- 1.2 Thesis Overview . . . 3

2 Background 5
- 2.1 Visual Question Answering . . . 5
- 2.2 Retrieval Augmented Generation . . . 6
  - 2.2.1 Retrieval . . . 7
  - 2.2.2 Knowledge-Based Visual Question Answering . . . 9
  - 2.2.3 Generation . . . 10
- 2.3 Reranking . . . 11
  - 2.3.1 Encoder Reranker . . . 12
  - 2.3.2 Sequence-to-Sequence Reranker . . . 14
  - 2.3.3 Modularized Reranker . . . 14
- 2.4 Conclusion . . . 15

3 Methodology 17
- 3.1 Benchmark Retriever . . . 17
- 3.2 Encoder Reranker . . . 20
  - 3.2.1 monoBERT . . . 20
  - 3.2.2 monoPreFLMR . . . 22
- 3.3 Sequence-to-Sequence Reranker . . . 23
  - 3.3.1 monoT5 . . . 23
  - 3.3.2 monoBLIP-2 . . . 24
- 3.4 Modularized Reranker . . . 26

3.4.1 MORES . . . 26
3.4.2 ModPreFLMR . . . 28
3.5 Listwise Ranking Loss . . . 29
3.5.1 Pointwise vs Listwise . . . 30
3.5.2 Listwise for Sequence-to-Sequence Rerankers . . . 31
3.5.3 Training on Retrieved Documents . . . 31
3.6 Conclusion . . . 32

4 Experiments **33**
4.1 Datasets . . . 33
4.1.1 OKVQA . . . 33
4.1.2 EVQA . . . 34
4.1.3 Difficulty of OKVQA vs EVQA . . . 34
4.2 Training Configuration . . . 36
4.3 Evaluation Metrics . . . 37
4.3.1 Recall . . . 37
4.3.2 McNemar Test . . . 38
4.3.3 Limitations of VQA Accuracy . . . 39
4.4 Conclusion . . . 40

5 Results **41**
5.1 Pointwise Results . . . 41
5.1.1 OKVQA Results . . . 41
5.1.2 EVQA Results . . . 42
5.1.3 Impact of Pre-Training . . . 43
5.1.4 Pointwise Loss Versus Recall . . . 45
5.2 Listwise Loss Function . . . 47
5.2.1 Listwise Loss Versus Recall . . . 48
5.2.2 Direct Prediction for Sequence-to-Sequence Rerankers . . . 50
5.3 Fine-Tuning on Retrieved Documents . . . 51
5.3.1 Fine-Tuning on Raw Retrieval . . . 52
5.3.2 Fine-Tuning on Upsampled Retrieval . . . 53
5.3.3 Fine-Tuning on Upsampled Retrieval with Listwise Loss . . . 54
5.4 Overall Results . . . 56
5.5 Ablation Studies . . . 57
5.5.1 Sigmoid Activation . . . 58
5.5.2 Encoder Vision Model . . . 58

5.5.3 Cross-Encoder Layers . . . 59
5.5.4 Freezing Vision Model . . . 60
5.5.5 Retrieval List Size . . . 61
5.6 Conclusion . . . 62

6 Conclusion **65**
6.1 Limitations . . . 66
6.2 Future Work . . . 66
6.2.1 Pre-Training Diversity . . . 66
6.2.2 Knowledge Distillation . . . 67

References **69**

Appendix A Tabular Results **77**
A.1 Pointwise Results . . . 77
A.2 Listwise Results . . . 78
A.3 Fine-Tuning on Retrieved Documents Results . . . 79
A.4 Ablation Results . . . 80

List of Figures

2.1 **Visual Question Answering (VQA)**: Involves answering questions about images, such as identifying details (e.g., eye color, objects), counting items (e.g., slices of pizza), and inferring context (e.g., actions and surroundings). (Antol et al., 2015) 5

2.2 **Retrieval Augmented Generation (RAG)**: A query retrieves relevant context from a database, which is then used by a generator to produce an accurate and contextually enriched answer. 6

2.3 **Dense Passage Retrieval (DPR)**: A question and a document are represented as dense vectors, and their relevance is determined using the dot product of these vectors. 8

2.4 **ColBERT**: A question and a document are represented as dense vectors, with relevance determined by computing the maximum dot product across token interactions. 9

2.5 **Cross Encoders**: A query and document are jointly encoded within a unified network, allowing for interactions throughout the entire network to compute similarity. 11

2.6 **Multi-Stage Reranking**: Initial retrieval uses cheaper features to filter irrelevant candidates, followed by reranking stages that employ more informative features to balance efficiency and accuracy for refined results. 12

3.1 **PreFLMR Retrieval**: The query consists of text, q, and an image, I, which is used to search for the most relevant text document, d. PreFLMR utilizes various network architectures to embed queries and documents and then leverages the ColBERT late-interaction mechanism to calculate similarity using token-level embeddings. (Lin et al., 2024b) 17

3.2 **monoBERT**: A BERT encoder cross-encodes a query and document, with the CLS token's embedding passed through an MLP to classify document relevance. 21

3.3 **monoPreFLMR**: Extends the PreFLMR query encoder to cross-encode a multimodal query and document, with the CLS embedding passed through an MLP for relevance classification 23
3.4 **monoT5**: Uses a pre-trained T5 as a cross-encoder to jointly encode the query and document with "true" or "false" tokens generated to predict document relevance. 24
3.5 **monoBLIP-2**: The query and document are concatenated as a prompt into BLIP-2 with "true" and "false" tokens generated to indicate document relevance. Either a Flan-T5 encoder-decoder structure (left) or OPT decoder-only (right) can be used. 26
3.6 **MORES**: Modularizes monoBERT into Representation Modules for independently encoding queries and documents and an Interaction Module for cross-encoding. 27
3.7 **Mid-Interaction** - Incorporates interaction earlier than late-interaction models like ColBERT but later than early-interaction models like monoPreFLMR and monoBLIP-2 28
3.8 **ModPreFLMR**: PreFLMR late-interaction embeddings from retrieval are passed into a small cross-encoder reranker containing either IB blocks (left) or BERT blocks (right). 29

4.1 **Comparison of OKVQA and EVQA:** OKVQA examples feature more general and everyday concepts, while EVQA examples involve more specialized and niche topics (Lin et al., 2024b) 35
4.2 **Reranker Parameter Sizes**: monoBLIP-2 has the most parameters, followed by monoPreFLMR and then ModPreFLMR. Despite its size, monoBLIP-2 has the fewest trainable parameters as it is fine-tuned with LoRA. 37
4.3 **PreFLMR-B Recall**: The first 1,040 questions in the test split are the most challenging to retrieve, as their recall is significantly lower than the overall recall, particularly for EVQA. 38

5.1 **OKVQA Recall@5 vs Efficiency (Pointwise Loss):** There is a clear trade-off between performance and efficiency. monoBLIP-2 Opt achieves the highest recall but has the longest processing time. In contrast, monoPreFLMR-B and ModPreFLMR offer quicker responses but with more modest recall. 42

5.2 **EVQA Recall@5 vs Efficiency (Pointwise Loss):** With the more challenging EVQA, monoBLIP-2 is the only reranker that improves performance over raw retrieval. Additionally, ModPreFLMR demonstrates both superior efficiency and performance compared to monoPreFLMR. 43
5.3 **Recall@5 vs Training Step (Pointwise Loss):** monoBLIP-2 reaches the highest performance at a much faster pace than monoPreFLMR and ModPreFLMR, highlighting the impact of extensive pre-training on model performance. . . . 45
5.4 **Pointwise Loss vs Recall (OKVQA ModPreFLMR IB):** There is no clear relationship between pointwise loss and recall during validation. 46
5.5 **Pointwise Loss vs Training Step (OKVQA ModPreFLMR IB):** The average loss of correctly ranked queries and incorrectly ranked queries are nearly identical. 46
5.6 **Pointwise Loss vs Logit (OKVQA ModPreFLMR IB):** Due to class imbalance, higher logits correspond to higher losses, regardless of the correctness of the ranking. 47
5.7 **Recall@5 vs Training Step (Listwise Loss):** When sampling noisy negatives from the full corpus, the listwise loss yields slightly worse performance than the pointwise loss. 48
5.8 **Listwise Loss vs Recall (OKVQA ModPreFLMR IB):** The listwise loss does not perfectly correlate with recall performance, though it fluctuates much less than the pointwise loss. 49
5.9 **Listwise Loss vs Training Step (OKVQA ModPreFLMR IB):** There is now a clear difference between the average loss of correctly ranked queries and incorrectly ranked queries. 49
5.10 **Listwise Loss vs Logit (OKVQA ModPreFLMR IB):** The value of the scores has a smaller impact on listwise loss than on pointwise loss, although significant enough to make it an unreliable indicator of recall. 50
5.11 **Recall@5 vs Loss Functions** (*the highlighted bar represents the best loss for the respective model*): The listwise loss performs slightly worse than the pointwise loss when training on noisy negatives sampled from the full corpus. Furthermore, for monoBLIP-2, training the pointwise loss with a single output consistently leads to better performance than using a text generation output. . 51
5.12 **Fine-tuning on Full Corpus vs Raw Retrieval** (*the highlighted bar represents the best sampling method for the respective model*): Despite a significant decrease in validation loss when fine-tuning on retrieved documents, improvements in recall vary among models. 52

5.13 **Recall@5 vs Training Step (Fine-Tuning on Raw Retrieval):** In general, there are performance improvements when fine-tuning on retrieved documents. However, for ModPreFLMR on EVQA, performance drops dramatically due to class imbalance. 53
5.14 **Recall@5 vs Training Step (Fine-Tuning on Upsampled Retrieval):** Sampling a fixed ratio of positive and negative samples when fine-tuning on retrieved documents addresses class imbalance and improves recall for EVQA. However, this significantly reduces the diversity of training samples, causing instability in OKVQA, which is a smaller dataset. 54
5.15 **Recall@5 vs Training Step (Fine-Tuning on Upsampled Retrieval with Listwise Loss):** Listwise loss is beneficial when there are highly confounding documents among the retrieved documents in small datasets like OKVQA, as it stabilizes training and improves performance. 55
5.16 **Recall@5 vs Sampling Method** (*the highlighted bar represents the best sampling method for the respective model*): For all rerankers, fine-tuning on retrieved documents leads to improvements in performance. However, it is important to ensure a balanced ratio of positive to negative documents when sampling. For OKVQA, it is better to use the listwise loss as there are highly confounding documents, whereas for EVQA with more diverse samples, it is better to use the pointwise loss as ground truth labels provide more direction. 56
5.17 **Sigmoid Activation Ablation:** Predicting using a single score with sigmoid activation improves training speed by reducing redundant parameters, compared to using two scores with softmax activation. 58
5.18 **Encoder Vision Model Ablation:** Removing the vision model worsens performance, although it is much worse on EVQA than OKVQA, highlighting the differing importance of visual features between these tasks. 59
5.19 **Cross-Encoder Layers Ablation:** A single cross-encoder layer for monoPreFLMR and 5 IB layers for ModPreFLMR are sufficient, with additional layers providing diminishing returns. 60
5.20 **Freezing Vision Model Ablation:** Freezing the vision model in monoPreFLMR improves performance, confirming that it is important to avoid overfitting. 60
5.21 **Recall@5 vs Retrieval List Size:** The performance of rerankers improves with more documents for powerful models like monoBLIP-2, while less powerful models perform optimally with less document due to introduced noise. 61

List of Tables

4.1 **Training Configurations:** We employ the AdamW optimizer for all models with an epsilon value of 10^{-8}, but with varying batch sizes, learning rates, and gradient accumulation strategies. Notably, monoBLIP-2 has a smaller batch size because it exceeds GPU memory capacity. 36

5.1 **Overall Results**: Rerankers that improve Recall@5 for PreFLMR-B retrieval can also consistently improve results across other recall positions and retrieval results. 57

A.1 **Appendix:** OKVQA Reranker Recall 77
A.2 **Appendix:** EVQA Reranker Recall 78
A.3 **Appendix:** Reranker Recall by Loss Function 78
A.4 **Appendix:** Fine-Tuning on Full Corpus vs Raw Retrieval 79
A.5 **Appendix:** Reranker Recall by Training Samples 79
A.6 **Appendix:** Reranker Recall by Activation Function 80
A.7 **Appendix:** Reranker Recall by Vision Model 80
A.8 **Appendix:** Reranker Recall by Cross-Encoder Layers 80
A.9 **Appendix:** Reranker Recall by Freezing Vision Model 80
A.10 **Appendix:** Recall@5 vs Retrieval List Size 81

Chapter 1

Introduction

Visual Question Answering (VQA) aims to build models that can answer questions about the content of images (Antol et al., 2015). Despite significant advancements in Visual Language Models (VLMs) like PaLI (Chen et al., 2022) and BLIP-2 (Li et al., 2023), datasets containing questions with niche content unlikely to be in the training data, such as OKVQA (Marino et al., 2019) and EVQA (Mensink et al., 2023), remain challenging. Knowledge-Based Visual Question Answering (KB-VQA) emerges as an extension of VQA to address this problem, borrowing from Retrieval Augmented Generation (RAG) by retrieving relevant information from external sources as support for generating accurate responses (Gui et al., 2021; Hu et al., 2023b; Lin et al., 2024a, 2022). Retrieval is a key aspect of RAG and has been well-studied to be fast and efficient using bi-encoder structures. These structures employ two separate encoders: one for documents, which can be pre-processed in advance, and another for queries, which are processed in real-time (Karpukhin et al., 2020). Relevance is computed using Maximum Inner Product Search (MIPS) (Shrivastava and Li, 2014) with Approximate Nearest Neighbor (ANN) algorithms, enabling quick and scalable retrieval from large knowledge bases (Johnson et al., 2019; Malkov and Yashunin, 2018). However, this efficiency may come at the expense of performance, which is critical, as VLMs are highly sensitive to prompts, and retrieving irrelevant information can decrease the quality of generated answers (Liu et al., 2024b).

This thesis explores reranking as a technique to improve retrieval by refining the initial list of retrieved documents. Compared to retrieval with bi-encoders, reranking is typically performed using slower, but more powerful cross-encoders that jointly encode queries and documents. While there has been substantial research on reranking information retrieval and text retrieval, there is a notable gap in applying these methods to KB-VQA. In information retrieval, multi-stage retrieval pipelines are common practice, where initial stages filter down information for more powerful rerankers to refine results (Matveeva et al., 2006). Text retrieval has recently seen significant developments with the adoption of multi-stage retrieval through

reranking, driven by advancements in pre-trained large language models like BERT (Nogueira and Cho, 2019) and T5 (Nogueira et al., 2020), that can be readily fine-tuned for assessing the relevance of retrieved documents. However, these advancements have not yet been explored for KB-VQA, where both visual and textual information need to be processed when reranking. This thesis takes advantage of recent advancements in large pre-trained VLMs to introduce reranking techniques for KB-VQA.

1.1 Contributions

Apart from the concurrent (non-peer reviewed) work of Wen et al. (2024), released in the same month as this thesis, we are the **first to propose multimodal rerankers for use in a multi-stage retrieval pipeline for KB-VQA.** Our rerankers demonstrate **statistically significant improvements in recall over state-of-the-art PreFLMR retrievers** (Lin et al., 2024b) on both OKVQA and EVQA. These improvements are consistent across the range of retrievers, from the smallest (PreFLMR-B) to the largest (PreFLMR-G). Our most powerful reranker, monoBLIP-2, uses BLIP-2 as the backbone and:

- Improves PreFLMR-G Recall@5 for EVQA from 32.7% to 39.9%. (Section 5.4)
- Improves PreFLMR-G Recall@5 for OKVQA from 27.5% to 47.5%. (Section 5.4)

We are also the **first to propose a reranker with a modularized approach, ModPreFLMR, that leverages computations from late-interaction retrieval.** This enables the use of a significantly smaller cross-encoder, serving as a starting point for what we introduce as a **mid-interaction mechanism** that strikes a middle ground in the performance-efficiency trade-off between late-interaction models like PreFLMR and early-interaction models like monoBLIP-2. Compared to monoBLIP-2, ModPreFLMR is 97x faster and still:

- Improves PreFLMR-B Recall@5 for OKVQA from 23.5% to 29.1%, surpassing PreFLMR-G with little computational overhead. (Section 5.3)

We also conduct extensive analysis on various mechanisms of training rerankers, including model configurations, sampling methods, and loss functions:

- We find that a common trend among the best reranker configurations is that they rely on strong pre-trained information and require fewer trainable parameters to prevent overfitting. (Section 5.1.3)

- We demonstrate that fine-tuning on the distribution of top retrieval candidates (Gao et al., 2021) significantly improves performance, however, it is important to sample a balanced ratio of positive to negative documents to address class imbalance. (Section 5.3)
- We elaborate on the theoretical advantages of training with a listwise loss that operates across a batch of documents (Gao et al., 2021; Zhuang et al., 2023) and demonstrate that, in practice, it leads to better results on small datasets with confounding documents, such as OKVQA. Conversely, a pointwise loss function that operates on individual query-document pairs proves more effective on larger datasets with diverse documents, such as EVQA. (Section 5.2)

In summary, we show that the best strategy for reranking is to leverage strong pre-trained models and then align them to the reranking task through a two-step process: first, training on diverse reranking datasets with a pointwise loss on individual query-document pairs; then, fine-tuning on specific downstream retrievers with a listwise loss on lists of documents.

1.2 Thesis Overview

Chapter 2 begins by diving into developments of KB-VQA and reranking in the literature. Chapter 3 details the three reranker architectures explored: monoPreFLMR, monoBLIP-2, and ModPreFLMR, and describes various training strategies. Chapter 4 describes the experiments conducted on OKVQA and EVQA, including training configurations and evaluation metrics. Chapter 5 presents results from the experiments, providing a comprehensive analysis of reranker performance. Finally, Chapter 6 our findings, discusses their limitations, and suggests avenues for future work.

Chapter 2

Background

This chapter provides background on the development of Knowledge-Based Visual Question Answering (KB-VQA), with an emphasis on retrieval. The chapter concludes by introducing reranking methods, including encoder-based, sequence-to-sequence, and modularized approaches, which form the foundation for this thesis.

2.1 Visual Question Answering

The objective of Visual Question Answering (VQA) is to interpret an image and provide accurate responses to questions related to its content. VQA can be traced back to the VQAv1 dataset (Antol et al., 2015), which includes images from MS-COCO (Lin et al., 2014) with open-ended questions and ground truth answers. Since then, many datasets of varying difficulties and contexts have been introduced, such as GQA (Hudson and Manning, 2019), Clevr-x (Salewski et al., 2020), and VQA-CP (Agrawal et al., 2018). Some datasets focus on specific domains, such as the medical domain in Med-VQA (Zhan et al., 2020).

Fig. 2.1 **Visual Question Answering (VQA)**: Involves answering questions about images, such as identifying details (e.g., eye color, objects), counting items (e.g., slices of pizza), and inferring context (e.g., actions and surroundings). (Antol et al., 2015)

Models used for VQA are Visual Language Models (VLMs), capable of processing images and text. Recent large-scale models, such as PaLI (Chen et al., 2022), leverage established unimodal pre-trained models like T5 and ViT as backbones. Similarly, Frozen (Tsimpoukelli et al., 2021) connects vision encoders to frozen language models via a lightweight mapping network, enabling rapid adaptation to new tasks. BLIP-2 (Li et al., 2023) introduces a Q-Former module to map frozen visual representations to frozen large language models.

There has also been considerable success in pre-training with masked modeling to take advantage of larger volumes of data. LXMERT (Tan and Bansal, 2019) integrates vision and language pre-training for better cross-modal reasoning. VisualBERT (Li et al., 2019) and VL-BERT (Su et al., 2019) adapt BERT-like architectures for joint vision-language representation learning. ViLBERT (Lu et al., 2019) separates visual and textual streams, interacting via co-attentional transformers. UNITER (Chen et al., 2020) unifies vision-language tasks through a single pretraining framework.

2.2 Retrieval Augmented Generation

A common source of error in VQA is when questions require detailed knowledge of specific categories or instances (Mensink et al., 2023). VLMs struggle to encode this long-tail information - rare details about niche topics - as it is seldom found in training data. This is also a significant issue with large language models (LLMs) in the text-only domain (Lewis et al., 2020). To address this, there has been substantial research into Retrieval-Augmented Generation (RAG), which retrieves relevant information from external knowledge corpora to support answer generation.

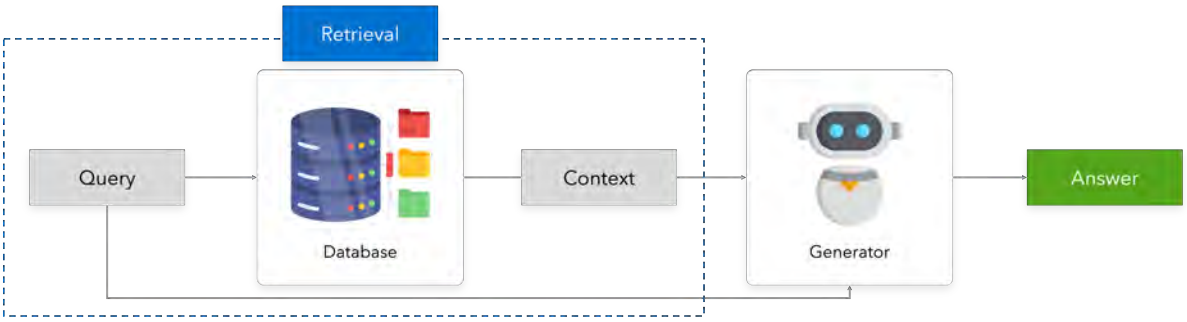


Fig. 2.2 **Retrieval Augmented Generation (RAG)**: A query retrieves relevant context from a database, which is then used by a generator to produce an accurate and contextually enriched answer.

2.2.1 Retrieval

The retrieval process begins with indexing, where a knowledge base is created from documents that can be used to support generation. Most knowledge bases for RAG are indexed in unstructured forms, consisting of raw text without storing any relationships between the different documents. Raw Wikipedia passages are commonly used in RAG research (Karpukhin et al., 2020; Lee et al., 2019; Lewis et al., 2020).

Given a query, q, retrieval calculates the probability that a document, d, is relevant (Equation 2.1) and selects those from the index with the highest probability.

$$s_{q,d} = P(\text{Relevant} = 1 \mid q, d) \tag{2.1}$$

Early work in retrieval, such as DrQA (Chen et al., 2017), used sparse retrieval techniques like TF-IDF and BM25 (Robertson et al., 1995; Sparck Jones, 1972), which utilize a lexicon to describe bag-of-words representations. These techniques focused on finding the most relevant documents by analyzing the structure of the documents and the distribution of words. As they are based on exact matches, they can use inverted list data structures for low-latency retrieval.

Later work advanced to dense retrieval techniques based on lower-dimensional vector spaces, which have been found to be more effective than sparse retrieval with high-dimensional word vectors. Dense retrieval techniques emphasize semantic similarity rather than exact words, with elements representing latent variables instead of words. Dense retrievers are bi-encoder architectures that generate two representations: one for the query, $\eta(q)$, and one for the document, $\eta(d)$, that can be compared with a similarity function, ϕ (Equation 2.2). This formulation is highly efficient because documents only need to be embedded into an index once, and during inference, only the queries need to be embedded for comparison.

$$s_{q,d} = \phi(\eta(q), \eta(d)) \tag{2.2}$$

BERT (Devlin et al., 2018) has driven the field of dense retrieval as the preferred encoder, η, due to its strong self-supervised pre-training. Systems like OrQA (Lee et al., 2019) and DPR (Karpukhin et al., 2020) leverage BERT and have been shown to outperform BM25 when trained on multiple documents. $s_{q,d}$ is maximized for relevant pairs using a cross-entropy loss on batches with negative sampling.

The choice of ϕ is often the dot product (Equation 2.3) and retrieval becomes a problem of Maximum Inner Product Search (MIPS) (Shrivastava and Li, 2014). MIPS can be performed very efficiently, particularly on GPUs, with Approximate Nearest Neighbor Search (ANN) algorithms. Methods based on hierarchical navigable small world (HNSW) graphs (Malkov and Yashunin, 2018) represent the current state of the art in ANN search. A popular open-

source library for ANN search is FAISS by Facebook (Johnson et al., 2019), which provides widespread accessibility to implementations of these techniques.

$$s_{q,d} = \eta(q)^T \eta(d) \tag{2.3}$$

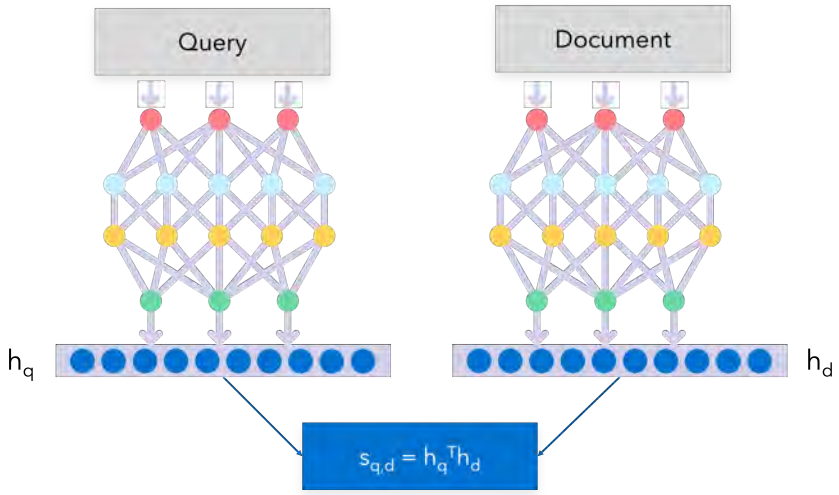


Fig. 2.3 **Dense Passage Retrieval (DPR)**: A question and a document are represented as dense vectors, and their relevance is determined using the dot product of these vectors.

Indeed, a trade-off with the computational efficiency of dense retrieval is that it condenses text into only a single embedding for comparison. An extension is ColBERT (Khattab and Zaharia, 2020), which preserves token-level embeddings and computes the dot product between all pairs of query and document tokens, summing across the maximum similarities of each query token. This method is more fine-grained than DPR with a single dot product, although it is still late-interaction, as similarities are only measured in the final layer. Typically, implementation is performed in a multi-stage process through a system like PLAID (Santhanam et al., 2022), where initial stages involve efficient candidate retrieval using centroid interaction and pruning techniques. This filters the results down before performing the full ColBERT search on the smaller set of filtered documents.

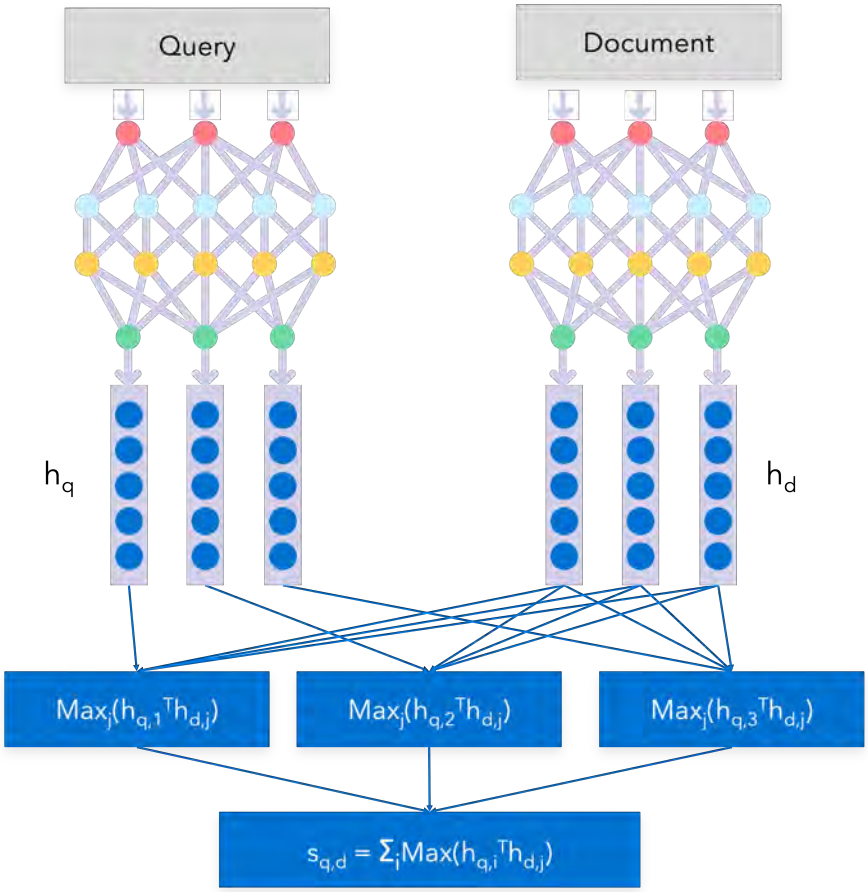


Fig. 2.4 **ColBERT**: A question and a document are represented as dense vectors, with relevance determined by computing the maximum dot product across token interactions.

2.2.2 Knowledge-Based Visual Question Answering

Knowledge-Based Retrieval-Augmented Generation (KB-VQA) is an extension of RAG to VQA, adapted to include visual information during retrieval. Several datasets have emerged recently for KB-VQA. OKVQA (Marino et al., 2019) is a dataset with human-annotated questions designed to require additional reasoning beyond what is directly in the image. However, performance on OKVQA has improved significantly even without retrieval, as modern large language models possess a considerable amount of internal information (Yang et al., 2022). A newer EVQA dataset (Mensink et al., 2023) has been deliberately constructed to contain questions with information that a model is unlikely to have internally. Existing VLMs that perform well on OKVQA struggle significantly with EVQA. However, an oracle model with access to relevant external documents has been shown to perform well, demonstrating that effective external knowledge retrieval is essential for success.

Most retrievers for KB-VQA are based on DPR, although they differ in the methodology for converting images and text into representations suitable for retrieval. The idea of CLIP (Radford et al., 2021b) forms the basis for bringing image and text embeddings into the same domain through training on large-scale text-image pairs using contrastive learning. REVIVE (Lin et al., 2022) and KAT (Gui et al., 2021) detect regions of the query image and use a CLIP model to search for the most relevant text in a knowledge base. REVEAL (Hu et al., 2023b) incorporates the query text as well as the query image for retrieval, using a VLM with a T5 and ViT backbone to jointly encode the query image and text, with a pooling layer to create a single dense representation for retrieval. Another approach is to utilize image-to-text transformations, such as captioning, to convert images into textual descriptions for purely text retrieval. RA-VQA (Lin and Byrne, 2022) and TRiG (Gao et al., 2022) convert images into multiple text formats to minimize information loss, including captions, object and attribute labels, and OCR text for DPR using text embeddings. Beyond DPR, FLMR (Lin et al., 2024a) and its extension PreFLMR (Lin et al., 2024b) are retrieval systems that measure similarities based on the ColBERT late interaction mechanism using token-level embeddings of images and text. Similar to REVEAL, FLMR employs a VLM with a BERT and ViT backbone to jointly encode the query image and text for retrieval.

2.2.3 Generation

Generation typically revolves around synthesizing the query and selected documents into a prompt as input into either a LLM for RAG (Guu et al., 2020; Lewis et al., 2020) or a VLM for KB-VQA (Lin et al., 2024a,b). However, a primary concern is that redundant information can interfere with generation, and overly long contexts can lead to the "Lost in the Middle" problem (Liu et al., 2024b). LLMs have been observed to focus only on the beginning and end of long texts, often forgetting the middle portion. Indeed, the quality of retrieved documents becomes particularly crucial when context sizes are limited, and only a few top documents can be utilized during generation. Many methods have been developed to address this. One such method is context selection or compression, such as RECOMP (Xu et al., 2023), which trains an information condenser, or LLMLingua (Jiang et al., 2023), which utilizes smaller language models like GPT-2 Small or LLAMA-7B to detect and remove unimportant tokens. Another approach is RAFT (Zhang et al., 2024) which fine-tunes LLMs using irrelevant documents. This process enables the model to learn when to disregard retrieved information during inference and instead generate responses based on its implicit knowledge.

2.3 Reranking

To improve retrieval, a more optimal method for computing the similarity between a query and a document goes beyond the late interactions of ColBERT and instead incorporates interactions throughout the entire network in what can be described as early interaction. In contrast to bi-encoders in DPR, which separately encode queries and documents, this class of models, known as cross-encoders, is designed to jointly process the query and document within a unified encoder (Equation 2.4).

$$s_{q,d} = \phi(q,d) \tag{2.4}$$

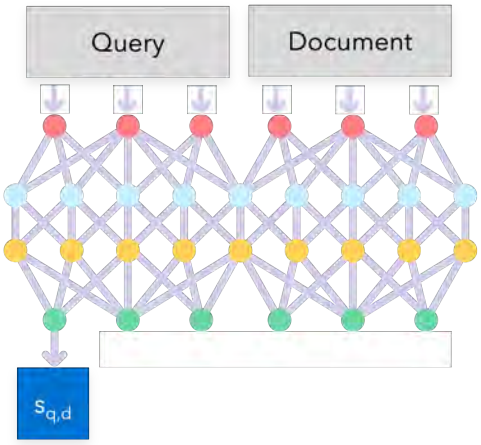


Fig. 2.5 **Cross Encoders**: A query and document are jointly encoded within a unified network, allowing for interactions throughout the entire network to compute similarity.

However, the major drawback of cross-encoders is their inability to leverage fast ANN computations as the similarity is no longer based on a simple dot product, making them unsuitable for scaling. Typically, cross-encoders are used after the initial retrieval results have been filtered down.

The idea of having multiple stages of reranking is common practice in information retrieval research due to its effectiveness in balancing efficiency and accuracy (Matveeva et al., 2006). Multi-stage ranking architectures are particularly prevalent in industry applications, including Alibaba's e-commerce search (Chen et al., 2023a), Baidu's web search (Zou et al., 2021), and YouTube's recommendation algorithm (Covington et al., 2016). Earlier stages use cheaper, less informative features like dense representations to discard irrelevant candidates. More expensive, informative features like cross-encoders are subsequently applied in later stages to refine rankings.

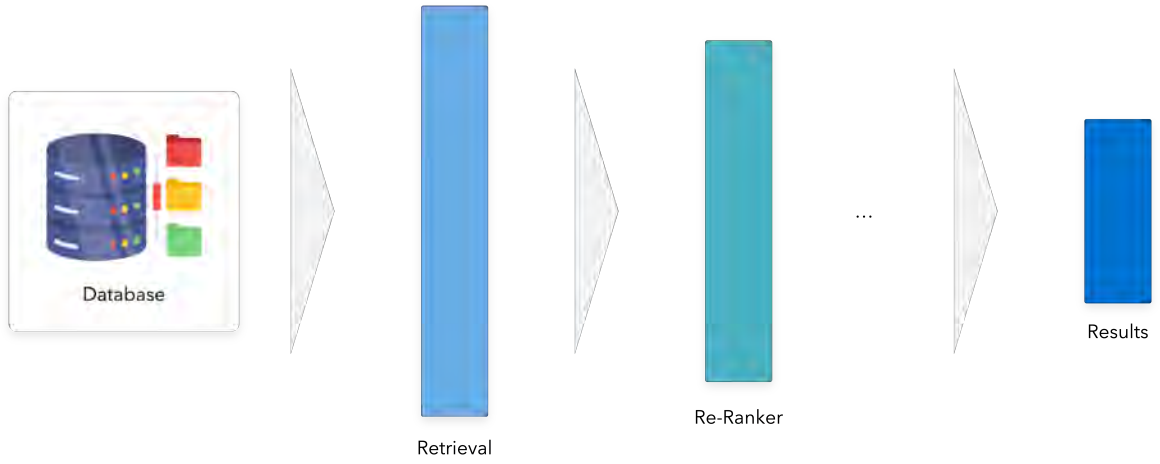


Fig. 2.6 **Multi-Stage Reranking**: Initial retrieval uses cheaper features to filter irrelevant candidates, followed by reranking stages that employ more informative features to balance efficiency and accuracy for refined results.

Indeed, this concept has been well-explored in the text domain. Already, the PLAID implementation of ColBERT itself can be considered a multi-stage reranking process. In initial stages, results are filtered using a computationally efficient nearest neighbor search. The final stage then applies the more sophisticated and resource-intensive ColBERT mechanism to a refined set of high-quality candidates.

More generally, there are three main types of text reranking architectures. The earliest form of reranking relies on leveraging pre-trained encoders such as BERT. However, more recently, practitioners have shifted towards using pre-trained sequence-to-sequence models like T5, which have demonstrated better performance. Additionally, some research has been conducted into modularized models that utilize pre-computed calculations to save computation by reusing calculations during ranking.

Apart from the concurrent (non-peer reviewed) work of Wen et al. (2024), released in the same month as this thesis, we are the first to extend reranking from the text domain to the multimodal domain of KB-VQA.

2.3.1 Encoder Reranker

The power of BERT's unsupervised pre-training has significantly influenced both retrieval and reranking domains. A notable application involves using the BERT architecture directly as a reranker following initial retrieval using BM25, an approach known as monoBERT (Nogueira and Cho, 2019). In this method, the query and document are concatenated and input into BERT, followed by a classification layer to determine document relevance. This approach's

effectiveness stems from its similarity to the Next Sentence Prediction (NSP) task used in BERT's original pre-training (Devlin et al., 2018).

The binary classification for relevance connects to the ranking task through the Probability Ranking Principle (Robertson and Belkin, 1978), which posits that documents should be ranked in decreasing order of their estimated relevance probability. All documents passing the retrieval stage are processed through the reranker and ranked in descending order of their logits. Training a reranker to predict this probability one query-document pair at a time is known as a pointwise approach, as each document is evaluated independently in the loss function. This method has the limitation that a document's score is independent of others in the query's result list. Indeed, optimizing a pointwise training loss does not theoretically guarantee optimal ranking performance, as the objectives of minimizing individual classification errors and achieving the best possible ordering of items are not aligned. However, empirical evidence shows that pointwise approaches often succeed in practice (Yates et al., 2021).

Nogueira et al. (2019) addresses this by introducing DuoBERT, which takes as input a query and two documents, outputting a prediction of which document is more relevant. This pairwise approach aims to minimize ranking inversions where document pairs are incorrectly ordered. The underlying rationale is that assessing documents in relation to each other, rather than in isolation, aligns more closely with the nature of ranking than predicting individual relevance scores. During inference, pairwise comparisons are made between all candidate documents, and scores are aggregated to produce the final ranking. There is a significant scalability difference between monoBERT and DuoBERT. MonoBERT can efficiently process thousands of input documents, as it scores each document independently. In contrast, DuoBERT requires comparing all possible document pairs, which scales quadratically with the number of documents, limiting it to an even later stage, typically handling only tens of documents. Hence, DuoBERT is typically employed following monoBERT in an additional reranking stage.

Listwise ranking losses have also been explored, which consider the entire document list during training, aligning even closer with the nature of ranking (Gao et al., 2021; Han et al., 2020; Ren et al., 2021). This approach learns to rank batches containing a single positive document and multiple negative documents. The model utilizes a listwise softmax function across the batch to maximize the positive document's score while reducing the scores of negative documents. Training is optimized using a cross-entropy loss function. Notably, this approach shares similarities with dense retrieval training, with both methods aiming to teach the model to assign higher ranks to positive examples and lower ranks to negative ones, effectively capturing the relative importance of documents within a given context.

2.3.2 Sequence-to-Sequence Reranker

Recent work in text reranking has found success in moving beyond using pre-trained encoders to pre-trained sequence-to-sequence models such as T5. T5 has been shown to adapt to a variety of natural language processing tasks beyond generation, including traditional classification tasks like sentiment analysis and topic classification, as well as regression tasks (Roberts and Raffel, 2020).

Nogueira et al. (2020) explored the application of the T5 model to text ranking, introducing a model called monoT5. Similar to monoBERT, monoT5 encodes the concatenation of the query and document but formats it as a prompt. This approach leverages internal model representations just before the generation of an output token for relevance classification. MonoT5 employs a pointwise approach to ranking with the decoder fine-tuned with a text generation loss to predict "true" if the document is relevant and "false" if it is not. During inference, post-processing is applied using a softmax function between the "true" and "false" token probabilities with the "true" token probability used to rerank the documents (Equation 3.20). Notably, experimental results indicate that monoT5 outperforms monoBERT with less training data, suggesting improved generalization capabilities.

A pairwise approach to ranking has also been extended to monoT5 through an implementation called duoT5 (Pradeep et al., 2021). In this model, the input consists of a concatenation of the query and two documents, formatted as a prompt. The model is fine-tuned to predict "true" if the first document is more relevant than the second, and "false" otherwise. Listwise ranking with the T5 model has also been explored by RankT5 (Zhuang et al., 2023), which employs a listwise softmax function. This method can no longer directly use T5 generation probabilities and requires modifications to produce numerical scores directly. Token probabilities were suitable for monoT5 and duoT5, as the pointwise loss and pairwise loss both train to predict probabilities. However, the listwise loss trains to predict raw scores, making the direct adaptation of token probabilities inappropriate.

We note that the reranker concurrently developed by (Wen et al., 2024) for VQA fits in the category of sequence-to-sequence rerankers. They use a pre-trained PaLI VLM as the backbone and train with a pairwise loss.

2.3.3 Modularized Reranker

The Modularized Reranking System (MORES) (Gao et al., 2020) is an extension of monoBERT that decouples the reranker into separate modules. Document embeddings in an intermediate layer are pre-computed and stored in an index, and during inference, query embeddings are calculated and then cross-encoded with the pre-computed document embeddings. Decoupling

the modules reduces expressiveness, as cross-encoding only begins after the query and document are first encoded separately. However, the efficiency benefits become apparent when we consider that the attention mechanism's complexity is quadratic with respect to the length of the input sequence. Consequently, cross-encoding has a complexity dependent on the lengths of both the query and document, represented as $N^{\text{Int}} \cdot (q+d)^2$, where N^{Int} is the number of cross-encoder layers, q is the query length, and d is the document length. By effectively separating the encoding of the query and document, where document embeddings are pre-computed and not modified, the complexity during inference becomes $N^{Q} \cdot q^2 + N^{\widehat{\text{Int}}} \cdot (q+d)^2$ where N^{Q} is the number of query encoding layers, and $N^{\widehat{\text{Int}}}$ is the number of interaction layers, which can be less than N^{Int} due to the pre-computed information. This separation allows for more efficient computation, especially when dealing with large document collections, as the document embeddings can be pre-computed and reused.

2.4 Conclusion

This chapter traced the development of VQA and highlighted the importance of KB-VQA to address questions that require niche information. We have explored how modern retrieval leverages MIPS with ANN to efficiently access relevant information from knowledge bases. The chapter then introduces the concept of reranking first-stage retrieval results with more computationally expensive but powerful cross-encoders. We trace the development of reranking from encoder-based methods to sequence-to-sequence models to modularized techniques with various training methodologies, including pointwise, pairwise, and listwise approaches. In the next chapter, we describe reranking in more technical detail and propose how they can be extended to incorporate images.

Chapter 3

Methodology

This chapter outlines our methodology for reranking KB-VQA retrieval results. We begin by describing the PreFLMR retriever (Lin et al., 2024b), which serves as the base for reranking. It retrieves text documents using a query composed of both image and text. Next, we describe our reranker architectures, drawing on established research in text-based reranking to inform our approach. We then look at specific strategies for improving the training process, including the use of listwise loss functions and training on retrieved document lists.

3.1 Benchmark Retriever

The baseline for retrieval is PreFLMR, designed for retrieving relevant documents for KB-VQA.

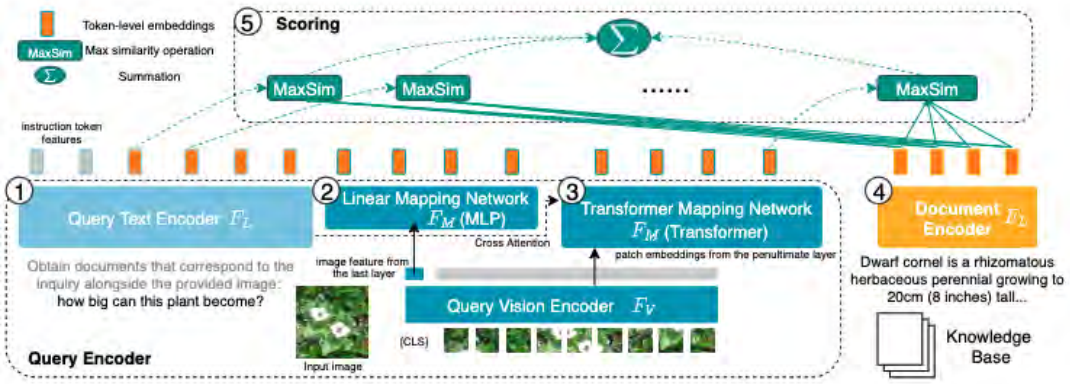


Fig. 3.1 **PreFLMR Retrieval**: The query consists of text, q, and an image, I, which is used to search for the most relevant text document, d. PreFLMR utilizes various network architectures to embed queries and documents and then leverages the ColBERT late-interaction mechanism to calculate similarity using token-level embeddings. (Lin et al., 2024b)

On the query side, a BERT text encoder, F_{L_Q} processes the text query, q. As BERT has a fixed number of positional embeddings, the length of the text, N_q is first truncated to a maximum of 512 tokens. The matrix of query embeddings $\hat{Q}_q$ is calculated with Equation 3.1 with dimensions, d_L, of 768. Subsequently, Equation 3.2 utilizes a one-layer Multi-Layer Perceptron (MLP), $F_{L_Q}^{MLP}$, to convert the embedding dimension to a lower dimension matrix Q_q with embedding size d_h, chosen to be 128.

$$\hat{Q}_q = F_{L_Q}(q) \in \mathbb{R}^{N_q \times d_L} \tag{3.1}$$

$$Q_q = F_{L_Q}^{MLP}(\hat{Q}_q) \in \mathbb{R}^{N_q \times d_h} \tag{3.2}$$

The image query, I, is processed by a ViT encoder, F_V. The dimension size, d_V, of ViT is 768. From the final layer, we utilize the single embedding of the CLS token as a holistic representation, capturing the overall features and characteristics of the entire image (Equation 3.3). We pass it through a 2-layer MLP mapping network, F_M^{MLP} which converts it into N_{vt} embeddings in the same embedding space, d_h, as the text embeddings (Equation 3.4). N_{vt} is chosen to be 32.

$$Q_{I,[CLS]} = F_V(I) \in \mathbb{R}^{1 \times d_V} \tag{3.3}$$

$$Q_I^{MLP} = F_M^{MLP}(Q_{I,[CLS]}) \in \mathbb{R}^{N_{vt} \times d_h} \tag{3.4}$$

Moreover, more fine-grained information about the image can be obtained from the N_v patch embeddings from the penultimate layer (Equation 3.5). N_v for ViT-B is 49 while for ViT-L and ViT-G, it is 256. These tokens are passed through an additional transformer module, F_M^{TR} which consists of N_M^{TR} transformer layers that cross-attend with the text query Q_q to obtain embeddings more relevant to the query text. The module ends with a single MLP Layer to map embeddings from d_v to d_h (Equation 3.6).

$$Q_{I,PATCH} = F_{V,-2}(I) \in \mathbb{R}^{N_V \times d_V} \tag{3.5}$$

$$Q_I^{TR} = F_M^{TR}(Q_{I,PATCH}, \hat{Q}_q) \in \mathbb{R}^{N_V \times d_h} \tag{3.6}$$

Equation 3.7 describes the aggregate token-level embeddings for the image and text query, Q, which is a concatenation of all the calculated text and image embeddings.

$$Q = \left[Q_q \quad | \quad Q_I^{MLP} \quad | \quad Q_I^{TR} \right] \in \mathbb{R}^{(N_q+N_V+N_{vt})\times d_h} = \mathbb{R}^{l_Q \times d_h} \tag{3.7}$$

On the document side, another BERT text encoder, F_{L_D} is used along with a single-layer MLP, $F_{L_D}^{MLP}$ to embed documents into a dimension size of d_h for the knowledge base (Equation 3.8). Similar to queries, the maximum context size of documents, N_d is set to 512.

$$D = F_{L_D}^{MLP}(F_{L_D}(d)) \in \mathbb{R}^{N_d \times d_h} \tag{3.8}$$

Ranking scores, $r_{q,d}$, between queries and documents are computed based on ColBERT's late-interaction mechanism (Santhanam et al., 2021), allowing each query token to interact with all document token embeddings as part of the calculation with the sum of the relevance of all queries token against the document token with the maximum relevance.

$$r_{q,d} = ColBERT(Q,D) = \sum_{i=1}^{l_Q} \max_{1\le j\le l_D} (Q_i^\top D_j) \in \mathbb{R}^{1\times 1} \tag{3.9}$$

The model is trained with a contrastive loss function (Equation 3.10) using in-batch negative sampling (Karpukhin et al., 2020), where with a batch size of N documents and N queries, there is a collection of 1 positive document, d^+, and $N-1$ negative documents, $N(q)$, per query.

$$L = -\sum_{(q,d^+)\in\mathscr{D}} \log \frac{\exp(r_{q,d^+})}{\exp(r_{q,d^+}) + \sum_{z\in\mathscr{N}(q)} \exp(r_{q,z})} \tag{3.10}$$

Due to the nature of training with a contrastive loss, ranking scores, $r_{q,d}$ do not have the same probabilistic interpretation as $s_{q,d}$ (Equation 2.1). However, they are representative of the degree of relevance between different documents and, hence, they should follow the same order, i.e., $r_{q,d_1} < r_{q,d_2} \iff s_{q,d_1} < s_{q,d_2}$

For inference, token-level document embeddings are indexed using the PLAID implementation (Santhanam et al., 2022). In this approach, each document is represented as a compact set of centroids obtained through k-means clustering. During retrieval, PLAID computes relevance scores between query vectors and these centroids. In the initial stages, low-relevance centroids are pruned early, significantly reducing the number of candidate documents for the final stage where the most promising candidates are fully decompressed and scored with the full ColBERT mechanism.

The text and image encoders of PreFLMR are initialized from pre-trained checkpoints of BERT and ViT. The authors experiment with models of various sizes, using text models from BERT-Small (28.8M), BERT-Medium (41.1M), BERT-Base (110M), to BERT-Large (340M),

and vision models from ViT-B (88M), ViT-L (303M) (Radford et al., 2021a), ViT-H (631M), and ViT-G (1.84B) (Cherti et al., 2023).

PreFLMR is trained on various datasets, starting with the MSMARCO (Bajaj et al., 2016) dataset using either the ColBERTv1 (Khattab and Zaharia, 2020) or ColBERTv2 (Santhanam et al., 2021) training scheme to properly initialize the parameters of the text encoder. This is followed by further training on various KB-VQA datasets including OKVQA (Marino et al., 2019), EVQA (Mensink et al., 2023), LLaVA (Liu et al., 2024a), OVEN (Hu et al., 2023a), WIT (Srinivasan et al., 2021), CC3M (Sharma et al., 2018), KVQA (Shah et al., 2019) and Infoseek (Chen et al., 2023b). Throughout this process, the vision encoder is kept frozen.

The authors find that varying the size of the BERT encoder does not significantly impact performance beyond BERT-Base. Similarly, scaling the size of the ViT encoder does not significantly improve performance beyond ViT-L. They release three checkpoints for the trained models on HuggingFace, all based on BERT-Base, but varying in ViT size: ViT-B[5], ViT-L[6], and ViT-G[7]. These models are hence classified as PreFLMR-B, PreFLMR-L, and PreFLMR-G, respectively, and serve as the retrieval results for reranking.

3.2 Encoder Reranker

The first method of reranking involves using a pre-trained encoder model architecture to jointly cross-encode a query and document with an additional classification head to represent their similarity.

3.2.1 monoBERT

Our model is inspired by monoBERT Nogueira et al. (2019), which is used for text-only reranking using a pointwise ranking objective on individual query-document pairs. The backbone is initialized from a pre-trained BERT model, serving as the cross-encoder, F_R, to jointly encode query, q, and document, d, as inputs. The format of the text input to the model, $T(q,d)$, begins with a CLS token, used as the final representation for relevance classification. This is followed by a concatenation of the query and document, separated and terminated by SEP tokens (Equation 3.11).

$$T(q,d) = [[CLS], q, [SEP], d, [SEP]] \tag{3.11}$$

[5] https://huggingface.co/LinWeizheDragon/PreFLMR_ViT-B
[6] https://huggingface.co/LinWeizheDragon/PreFLMR_ViT-L
[7] https://huggingface.co/LinWeizheDragon/PreFLMR_ViT-G

For BERT, the prompt is tokenized with the WordPiece tokenizer (Devlin et al., 2018), and learned token embeddings are used as inputs into the cross-encoder layers. Learned positional embeddings are added to distinguish the sequential positions of the tokens, up to a maximum context length of 512 tokens for BERT. This means that tokens beyond this limit are truncated, which may be a concern for some queries and documents that exceed this length. Finally, learned segment embeddings are added to differentiate between tokens representing the query and those representing the document.

The output of the cross-encoder is token embeddings, $H_{q,d}$, with a dimension size, d_L, of 768. The embedding of the CLS token is then passed through a single-layer MLP, F_R^{MLP}, to classify the probability that the document is relevant to the query. Two logits are produced, $z_{q,d}$, one representing the probability of being relevant and the other representing the probability of being not relevant. A softmax activation function is applied to ensure that the two probabilities sum to 1 (Equation 3.12).

$$\begin{aligned} Q_{q,d} &= F_R(T(q,d)) \in \mathbb{R}^{N_q \times d_L} \\ z_{q,d} &= F_R^{MLP}(Q_{q,d,CLS}) \in \mathbb{R}^{1\times 2} \\ s_{q,d} &= \text{Softmax}(z_{q,d})_0 \in \mathbb{R}^{1\times 1} \end{aligned} \tag{3.12}$$

The model is trained in a pointwise approach across each query-document pair using a pointwise binary cross-entropy loss (Equation 3.13), where D_{pos} is the set of relevant documents and D_{neg} is the set of non-relevant documents.

$$L = -\sum_{d\in D_{\text{pos}}} \log(s_{q,d}) - \sum_{d\in D_{\text{neg}}} \log(1-s_{q,d}) \tag{3.13}$$

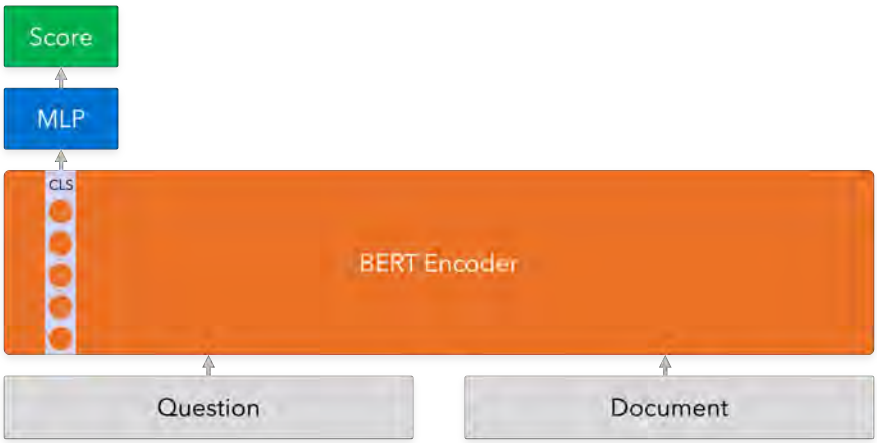


Fig. 3.2 **monoBERT**: A BERT encoder cross-encodes a query and document, with the CLS token's embedding passed through an MLP to classify document relevance.

3.2.2 monoPreFLMR

To extend the encoder approach to the multimodal domain, we require an encoder that can jointly embed images and texts. A natural choice is the PreFLMR query encoder (Equation 3.7), which can be repurposed from retrieval to reranking as it has already been pre-trained to embed images and texts in the context of computing relevance. A reranker using PreFLMR as a backbone can similarly be named monoPreFLMR.

A necessary modification is that the BERT text embeddings, Q_q, can no longer just contain the query but now must also contain the target document, $Q_{q,d}$. This is naturally addressed by modifying the input to contain the concatenation of the document as well (Equation 3.14). As the text embeddings are also based on a BERT model, we can use the same input format, truncation length, and segment embeddings as monoBERT (Equation 3.11).

$$\begin{aligned} \hat{Q}_{q,d} &= F_{L_Q}(T(q,d)) \in \mathbb{R}^{N_q \times d_L} \\ Q_{q,d} &= F_{L_Q}^{MLP}(\hat{Q}_{q,d}) \in \mathbb{R}^{N_q \times d_h} \end{aligned} \tag{3.14}$$

However, further modifications need to be made as PreFLMR text embeddings do not have any mechanism to attend to image embeddings. Hence, it is necessary to introduce additional cross-encoder layers that incorporate all the output embeddings of the PreFLMR query encoder, both text and image, for further processing to consolidate information. Particularly, we use a stack of N_R transformer layers, F_R, so that information from image embeddings can be incorporated in the resulting CLS embedding of $H_{q,d}$ (Equation 3.15).

$$H_{q,d} = F_R(\left[Q_{q,d} \quad | \quad Q_I^{MLP} \quad | \quad Q_I^{TR} \right]) \in \mathbb{R}^{l_Q \times d_h} \tag{3.15}$$

This CLS embedding is then similarly passed through a single-layer MLP, F_R^{MLP}, for pointwise relevance classification. However, we modify monoBERT to predict a single output using a sigmoid activation function (Equation 3.16), instead of the traditional two outputs with a softmax activation. Empirical results in Section 5.5.1 demonstrates that this adjustment enhances performance.

$$s_{q,d} = \sigma(F_R^{MLP}(H_{q,d,CLS})) \in \mathbb{R}^{1 \times 1} \tag{3.16}$$

Similar to monoBERT, the monoPreFLMR is also trained in a pointwise manner with a binary cross-entropy loss (Equation 3.13).

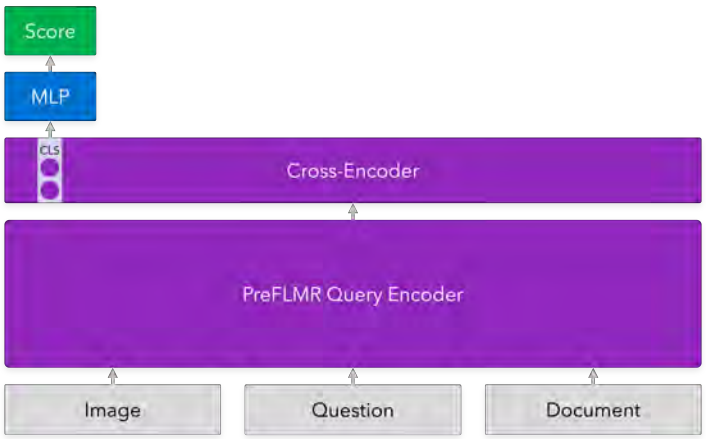


Fig. 3.3 **monoPreFLMR**: Extends the PreFLMR query encoder to cross-encode a multimodal query and document, with the CLS embedding passed through an MLP for relevance classification

3.3 Sequence-to-Sequence Reranker

The second method of reranking involves using a pre-trained sequence-to-sequence model architecture. In this approach, the model is prompted with the query and document, and relevance is assessed based on the probability of generating a target token.

3.3.1 monoT5

Our model is inspired by monoT5 (Nogueira et al., 2020), the successor of monoBERT. The backbone is initialized from a pre-trained T5 model, similarly serving as the cross-encoder, F_R that jointly encodes the query, q and document, d. The format of the input to the model, $T(q,d)$ is modified to align more with the prompt of a sequence-to-sequence task (Equation 3.17). It features verbatim text that labels the query, q and document, d, followed by the text "relevant" which prompts the model to assess the relevancy of the generated output.

$$T(q,d) = \text{Query: } q \text{ Document: } d \text{ Relevant:} \tag{3.17}$$

For the T5 model, joint encoding is performed by the encoder, F_R^{Enc}, and predictions are made by the decoder, F_R^{Dec}, in the form of token logits, $z_{q,d}$, where V represents the size of the vocabulary (Equation 3.18). The model is trained to generate the token immediately after the prompt, w, as either "true" or "false," indicating whether the document is relevant to the query. The choice of target text is not critical, provided it is represented by single tokens in the tokenizer.

$$z_{q,d} = F_R^{Dec}(F_R^{Enc}(T(q,d))) \in \mathbb{R}^{1\times V} \tag{3.18}$$

Training is pointwise and performed on individual query-document pairs using a text-generation cross-entropy loss to maximize the probability of "true" for positive documents and the probability of "false" for negative documents (Equation 3.19).

$$\begin{aligned} L &= -\sum_{d\in D_{\text{pos}}} \log(s_{q,d}) - \sum_{d\in D_{\text{neg}}} \log(1-s_{q,d}) \\ &= -\sum_{d\in D_{\text{pos}}} \log(p(w=\text{true})) - \sum_{d\in D_{\text{neg}}} \log(p(w=\text{false})) \\ &= -\sum_{d\in D_{\text{pos}}} \log\left(\frac{\exp(z_{q,d}^{(\text{true})})}{\sum_i \exp(z_{q,d}^{(i)})}\right) - \sum_{d\in D_{\text{neg}}} \log\left(\frac{\exp(z_{q,d}^{(\text{false})})}{\sum_i \exp(z_{q,d}^{(i)})}\right) \end{aligned} \tag{3.19}$$

During inference, the probability of relevance is assessed by comparing the normalized probabilities of only the "true" and "false" tokens. A softmax function is applied to their logits, and the resulting probability of the "true" token is taken as the relevance probability (Equation 3.20).

$$s_{q,d} = \frac{\exp(z_{q,d}^{(\text{true})})}{\exp(z_{q,d}^{(\text{true})}) + \exp(z_{q,d}^{(\text{false})})} \in \mathbb{R}^{1\times 1} \tag{3.20}$$

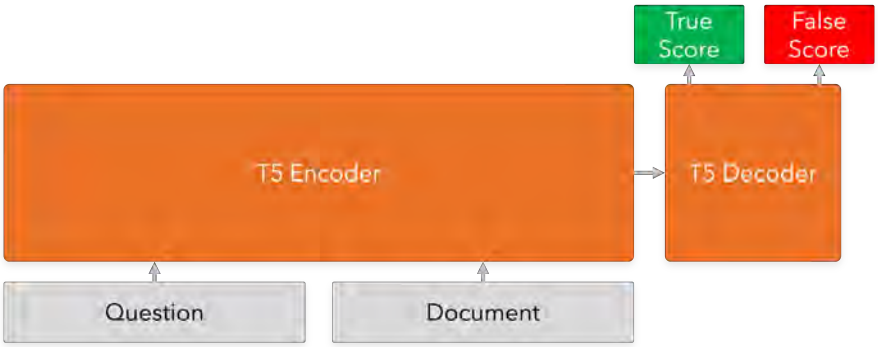


Fig. 3.4 **monoT5**: Uses a pre-trained T5 as a cross-encoder to jointly encode the query and document with "true" or "false" tokens generated to predict document relevance.

3.3.2 monoBLIP-2

To extend the sequence-to-sequence approach to the multimodal domain, we require a sequence-to-sequence model that can jointly embed images and texts. A natural choice is a VLM that can

process both images and texts to generate output, such as BLIP-2, which has been employed by Lin et al. (2024b) for generation after PreFLMR retrieval. A reranker using BLIP-2 as a backbone can similarly be named monoBLIP-2.

The BLIP-2 model uses a frozen image encoder to process the input image, which is then passed through a trainable Q-Former module to extract a fixed number of output embeddings. These image embeddings are subsequently concatenated with accompanying text embeddings and fed into a frozen language model for generation. The placement of image embeddings as tokens at the beginning of the input sequence is independent of the rest of the inputs meaning that we can precisely follow the monoT5 prompt format (Equation 3.17) as well as use a pointwise loss with "true" or "false" text generation targets, Equation 3.19.

The BLIP-2 image encoder is always a pre-trained ViT model, which remains frozen during training. For the BLIP-2 language model, we explore two alternatives: the decoder-only OPT model (Zhang et al., 2022) and the encoder-decoder Flan-T5 model (Chung et al., 2024). When using Flan-T5, predictions follow the method described in Equation 3.18, however, predictions using OPT follow Equation 3.21 where the decoder, F_R^{Dec}, handles both cross-encoding and prediction.

$$z_{q,d} = F_R^{Dec}(T(q,d)) \in \mathbb{R}^{1\times V} \tag{3.21}$$

While only the Q-Former is trained during BLIP-2 pre-training, when adapting to a specific downstream task such as predicting text relevance, we can follow the methodology of Lin et al. (2024a) by fine-tuning all the parameters of the model, including those of the language and image models. This approach allows for flexibility in adapting to the task. However, Low-Rank Adaptation (LoRA) is necessary, a method that reduces the number of modified parameters and regularizes the fine-tuning process to prevent catastrophic forgetting Hu et al. (2021). LoRA achieves this by decomposing the weight update, $\Delta W \in \mathbb{R}^{d\times k}$, into low-rank matrices, BA, where $A \in \mathbb{R}^{r\times k}$ and $B \in \mathbb{R}^{d\times r}$, and $r \ll \min(d,k)$. This decomposition significantly reduces the complexity and enhances the efficiency of the adaptation. The weight update of parameters is then scaled by α so that the new weights are $W' = W + \frac{\alpha}{r}BA$. This decomposition allows the model to retain essential features from the pre-trained state while efficiently learning the task-specific parameters.

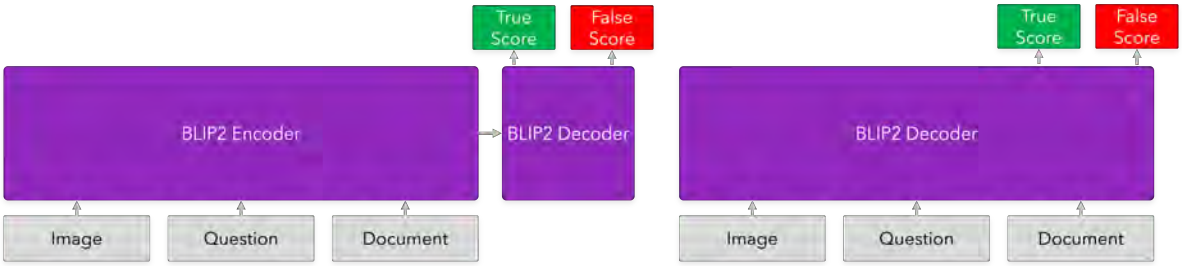


Fig. 3.5 **monoBLIP-2**: The query and document are concatenated as a prompt into BLIP-2 with "true" and "false" tokens generated to indicate document relevance. Either a Flan-T5 encoder-decoder structure (left) or OPT decoder-only (right) can be used.

3.4 Modularized Reranker

The third method of reranking uses PreFLMR late-interaction embeddings from retrieval as inputs to a small cross-encoder in the reranker. We treat retrieval and reranking as separate modules which share computations for efficiency. We introduce this as a concept of mid-interaction where the interaction stage is earlier than late-interaction models like ColBERT but later than early-interaction cross-encoder rerankers.

3.4.1 MORES

Our model is inspired by MORES (Gao et al., 2020), which speeds up ranking by first modularizing the process into two BERT-based Representation Modules, F_{L_Q} and F_{L_D}, which independently encode query and document tokens into dense embeddings (Equation 3.22).

$$\begin{aligned} Q &= F_{L_Q}(q) \in \mathbb{R}^{N_q \times d_L} \\ D &= F_{L_D}(d) \in \mathbb{R}^{N_d \times d_L} \end{aligned} \tag{3.22}$$

Subsequently, a cross-encoder Interaction Module, denoted as F_R, utilizes these embeddings as inputs to generate a relevance score. The Interaction Module comprises Interaction Blocks (IBs), F_{IB}, which are modified versions of the standard BERT layer architecture. IBs perform a query-to-document cross-attention operation, F_{IB}^{X-Attn}, followed by query self-attention, $F_{IB}^{Self-Attn}$ (Equation 3.23). Similar to BERT, the output culminates in a single-layer MLP, F_{IB}^{MLP}, but applied only to query token embeddings. Operations within a block include layer normalization (LN) and residual connections.

$$\begin{aligned} Q_x &= \text{LN}(F_{IB}^{X-Attn}(Q,D)+Q) \in \mathbb{R}^{N_q \times d_L} \\ Q_{\text{self}} &= \text{LN}(F_{IB}^{Self-Attn}(Q_x,Q_x)+Q_x) \in \mathbb{R}^{N_q \times d_L} \\ F_{IB}(Q,D) &= \text{LN}(F_{IB}^{MLP}(Q_{\text{self}})+Q_{\text{self}}) \in \mathbb{R}^{N_q \times d_L} \end{aligned} \tag{3.23}$$

Employing a stack of IB layers refines the hidden query token representations, Q, through multiple rounds of interactions while keeping the document representation, D, unchanged (Equation 3.24). This approach offers an advantage over monoBERT's heavy full-attention mechanism applied to the concatenated query-document sequence, which has a complexity of $(d+q)^2$. Instead, the IB layers reduce the complexity to q^2 for query self-attention followed by qd for query-document cross-attention. In essence, attention is being performed on just the query.

$$F_R(Q,D) = F_{IB_N}\left(F_{IB_{N-1}}\left(\ldots F_{IB_2}\left(F_{IB_1}(Q,D)\right)\ldots\right)\right) \in \mathbb{R}^{N_q \times d_L} \tag{3.24}$$

Similar to monoBERT, to measure relevance, the CLS token's embedding from the final IB layer is passed through a single-layer MLP, F_R^{MLP}, to classify the probability that the document is relevant to the query (Equation 3.25).

$$s_{q,d} = \text{Softmax}(F_R^{MLP}(F_R(Q,D)_{CLS})) \in \mathbb{R}^{1\times 1} \tag{3.25}$$

MORES trains all three transformer modules jointly with a pointwise ranking loss on query-document pairs (Equation 3.13). When training is complete, the three modules are decoupled, stored separately, and applied at the desired time. The benefit of this approach is that document embeddings can be pre-computed and stored in an index ahead of inference. During inference, the query embedding is computed and cross-encoded with document embeddings using the interaction module to determine the ranking.

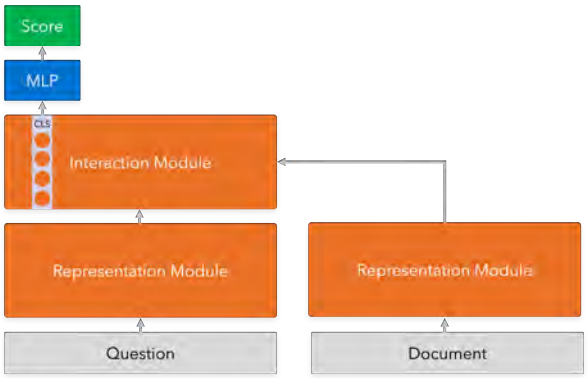


Fig. 3.6 **MORES**: Modularizes monoBERT into Representation Modules for independently encoding queries and documents and an Interaction Module for cross-encoding.

3.4.2 ModPreFLMR

Drawing inspiration from MORES, a modularized approach to multimodal reranking can be developed using PreFLMR retrieval encoders as the representation module. These encoders, already trained to measure relevance, provide query embeddings (Equation 3.7) and document embeddings (Equation 3.8) from the retrieval stage. A small cross-encoder, F_R, can then be used as the interaction module during reranking, taking these embeddings as inputs. To output the necessary probability of relevance, a single-layer MLP, F_R^{MLP}, with a sigmoid activation function is used (Equation 3.26). A modularized approach using PreFLMR can be named ModPreFLMR.

$$s_{q,d} = \sigma(F_R^{MLP}(F_R(\left[Q_q \quad | \quad Q_I^{MLP} \quad | \quad Q_I^{TR} \quad | \quad D\right])_{CLS})) \in \mathbb{R}^{1\times 1} \tag{3.26}$$

Effectively, this approach replaces the ColBERT late-interaction of PreFLMR (Equation 3.9) with a more sophisticated yet relatively small cross-encoder interaction module, F_R. This change theoretically enhances expressiveness by shifting the interaction stage earlier to a midway point, though not as early as early-interaction rerankers like monoPreFLMR and monoBLIP-2.

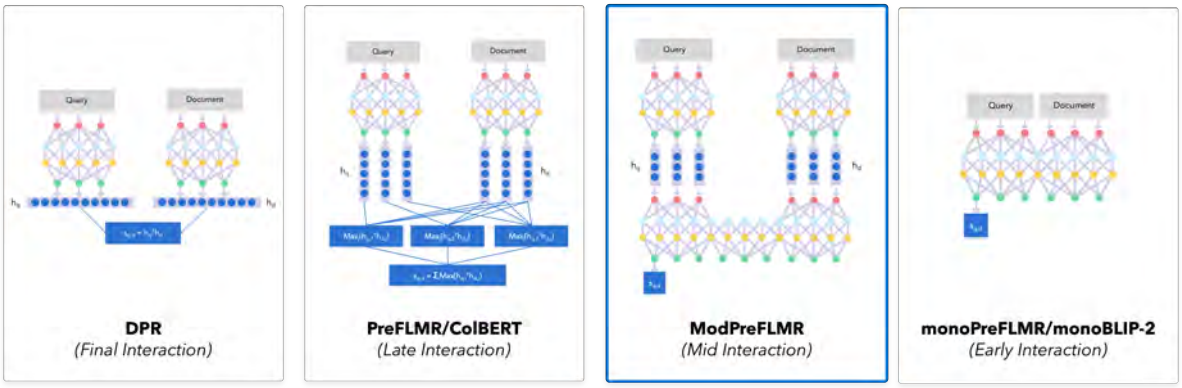


Fig. 3.7 **Mid-Interaction** - Incorporates interaction earlier than late-interaction models like ColBERT but later than early-interaction models like monoPreFLMR and monoBLIP-2

For F_R, we experiment with two approaches: a stack of IB modules with self-attention across query embeddings and cross-attention across document embeddings (Equation 3.24[8]), and a stack of vanilla BERT modules (Devlin et al., 2018) with self-attention across both query and document embeddings. Notably, the trade-off in computational complexity is evident in the size of their output dimensions (Equation 3.27).

[8]Implementation based on https://github.com/luyug/MORES/blob/dev/bert_mores.py

$$\begin{aligned} BERT\left(\left[Q_q \mid Q_I^{MLP} \mid Q_I^{TR} \mid D\right]\right) &\in \mathbb{R}^{(l_Q+N_d)\times d_h} \\ IB\left(\left[Q_q \mid Q_I^{MLP} \mid Q_I^{TR} \mid D\right]\right) &\in \mathbb{R}^{l_Q\times d_h} \end{aligned} \tag{3.27}$$

Unlike MORES, we only train the cross-encoder interaction module, F_R, and the MLP, F_R^{MLP}, essentially freezing the representation modules. This is necessary for ModPreFLMR as the representation modules are shared with PreFLMR retrieval, and we do not want to impact the retrieval performance. During inference, it becomes apparent that ModPreFLMR is more efficient than MORES, as we take advantage of embeddings pre-computed from the PreFLMR retrieval stage. This means we neither need to store document embeddings in an index beforehand nor compute query embeddings.

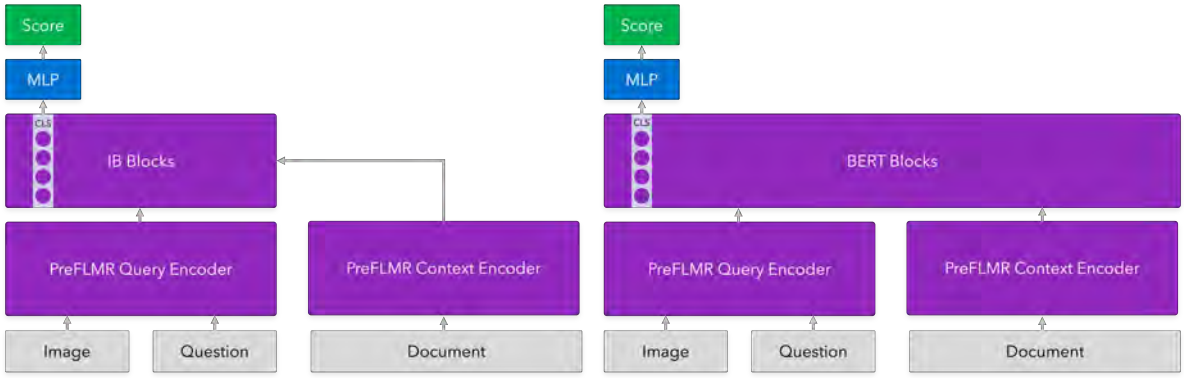


Fig. 3.8 **ModPreFLMR**: PreFLMR late-interaction embeddings from retrieval are passed into a small cross-encoder reranker containing either IB blocks (left) or BERT blocks (right).

3.5 Listwise Ranking Loss

Currently, our approaches are trained with a pointwise ranking loss on individual query-document pairs where a relevance score, $s_{q,d}$, is learned to represent the probability that a document is relevant. An alternative approach explores listwise ranking losses (Gao et al., 2021; Zhuang et al., 2023), which instead trains on a batch of retrieved documents and learns to predict relevance scores, $r_{q,d}$, that are higher for positive documents and lower for negative documents in the result list.

The listwise softmax function (Equation 3.28) is trained on batches containing one positive document, d^+, and N negative documents, $\mathcal{N}(q)$, per query:

$$L = -\sum_{(q,d^+)\in\mathcal{D}} \log \frac{\exp(r_{q,d^+})}{\exp(r_{q,d^+}) + \sum_{z\in\mathcal{N}(q)} \exp(r_{q,z})} \tag{3.28}$$

3.5.1 Pointwise vs Listwise

By analyzing gradients, we demonstrate why a listwise loss is more aligned with the ranking task. We assume there is a single query in the batch, B, with one positive document d^+ and N negative documents, totalling $1+N$ samples.

For the pointwise loss with a sigmoid activation, the gradient with respect to a network weight W is derived in Equation 3.29.

$$\begin{aligned} L &= -\log\left(\sigma(z_{q,d^+})\right) - \sum_{d\in B\setminus d^+} \log\left(1-\sigma(z_{q,d})\right) \\ \frac{\partial L}{\partial W} &= \frac{\partial z_{q,d^+}}{\partial W}\left(\sigma(z_{q,d^+})-1\right) + \sum_{d\in B\setminus d^+} \frac{\partial z_{q,d}}{\partial W}\sigma(z_{q,d}) \\ &= \frac{\partial z_{q,d^+}}{\partial W}\left(s_{q,d^+}-1\right) + \sum_{d\in B\setminus d^+} \frac{\partial z_{q,d}}{\partial W} s_{q,d} \end{aligned} \tag{3.29}$$

where $s_{q,d} = \sigma(z_{q,d})$.

For the listwise softmax loss, the gradient with respect to a network weight W is derived in Equation 3.30.

$$\begin{aligned} L &= -\log\left(\frac{\exp(z_{q,d^+})}{\sum_{d\in B}\exp(z_{q,d})}\right) \\ \frac{\partial L}{\partial W} &= \frac{\sum_{d\in B}\frac{\partial z_{q,d}}{\partial W}\exp(z_{q,d})}{\sum_{d\in B}\exp(z_{q,d})} - \frac{\partial z_{q,d^+}}{\partial W} \\ &= \frac{\partial z_{q,d^+}}{\partial W}\left(s_{q,d^+}-1\right) + \sum_{d\in B\setminus d^+} \frac{\partial z_{q,d}}{\partial W} s_{q,d} \end{aligned} \tag{3.30}$$

where $s_{q,d} = \frac{\exp(z_{q,d})}{\sum_{d^+\in B}\exp(z_{q,d^+})}$.

Indeed, the form of the gradient with respect to the scores $s_{q,d}$ is the same for both pointwise and listwise losses. In both cases, weights are updated based on the direction of documents with poorly predicted scores. However, the key difference lies in the calculation of the scores. In the pointwise case, the score is calculated independently of other documents in the batch, and so the gradient represents how well it aligns with ground truth labels. In contrast for a listwise loss, the score is dependent on the batch, and so the gradient represents how much the positive document surpasses the negative documents. In this sense, the listwise loss aligns more with ranking as the gradient considers the relative ranking between positive and negative documents in contrast to the pointwise loss which considers only the score of the individual document.

3.5.2 Listwise for Sequence-to-Sequence Rerankers

The outputs of monoPreFLMR and ModPreFLMR are singular numeric scores, $\mathbb{R}^{1\times1}$, and can be immediately repurposed for the softmax function of the listwise loss. However, the output of monoBLIP-2 is a logit over the entire vocabulary, $\mathbb{R}^{1\times V}$, and this output requires a softmax over the "true" and "false" tokens that cannot be readily repurposed for the listwise loss. Indeed, it is possible to just use the logit of the "true" token; however, this would require constraining the "false" token to always be 0 and excluding all other tokens during training (Zhuang et al., 2023). A more suitable approach is to use the unnormalized logits of an unused token, `<extra_id_10>` and taking it as the ranking score (Equation 3.31). This logit is directly used to train the listwise loss.

$$r_{q,d} = z_{q,d}^{(\texttt{<extra\_id\_10>})} \in \mathbb{R}^{1\times1} \tag{3.31}$$

By configuring monoBLIP-2 to generate a single output, we can also experiment training with a pointwise loss using a sigmoid activation (Equation 3.32) rather than relying on a text generation loss function.

$$s_{q,d} = \sigma(z_{q,d}^{(\texttt{<extra\_id\_10>})}) \in \mathbb{R}^{1\times1} \tag{3.32}$$

3.5.3 Training on Retrieved Documents

If we train on documents from the full corpus, the reranker may not generalize as well during inference, since retrieval results are filtered and do not follow the same distribution as the rest of the corpus. Gao et al. (2021) demonstrates that sampling negative documents from retrieval results is less noisy than sampling from the full corpus, and helps improve recall performance.

However, the challenge with training on the top portion of retriever results is that they are more homogeneous, having passed through the same retrieval filter, so they are more likely to share confounding characteristics. Gao et al. (2021) suggests that a listwise loss is especially advantageous in this case. Particularly on a dataset with multiple confounding features, it may be easier for a pointwise loss to predict similar scores, as this could result in a good average cross-entropy loss against ground truth labels. However, a listwise loss would be punished for predicting similar scores and would be more motivated to identify features that can distinguish positive documents from negative ones.

3.6 Conclusion

This chapter presented our methodology reranking in the multimodal domain, specifically aimed at improving retrieval results for VQA. We began by describing the PreFLMR retriever as our baseline for reranking before describing three reranking approaches:

- **Encoder Reranker (monoPreFLMR)** - We fine-tune a pre-trained PreFLMR query encoder as a cross-encoder, taking in both the query and the document and using their joint encoding to predict relevance.
- **Sequence-to-Sequence Reranker (monoBLIP-2)** - We fine-tune a pre-trained BLIP-2 model as a cross-encoder, taking in a prompt containing the query and document and using the generated token probabilities to predict relevance.
- **Modularized Reranker (ModPreFLMR)** - We reuse PreFLMR late-interaction embeddings from the retrieval stage in a mid-interaction mechanism where a lightweight cross-encoder is sufficient for predicting relevance.

We then suggest two strategies for improving reranker training:

- **Listwise Loss** - Trains the model to rank documents relative to one other as opposed to a pointwise loss which learns individual scores.
- **Training on Retrieved Documents** - Teaches the model about the distribution of top retrieval results which is very different from rest of the corpus.

The next chapter will detail experiments to evaluate these methodologies, including dataset descriptions, training configurations, and evaluation metrics.

Chapter 4

Experiments

This chapter details the experiments conducted on two KB-VQA datasets: OKVQA (Marino et al., 2019) and EVQA (Mensink et al., 2023). We start by comparing and highlighting their differences. We then discuss the training configurations for rerankers, including optimizer parameters and batch strategies. Finally, we describe the recall metric used for evaluation and discuss its implications on VQA Accuracy.

4.1 Datasets

Our experiments focus on two common KB-VQA datasets: OKVQA and EVQA. We utilize the implementation based on the M2KR benchmark by Lin et al. (2024b), which allows for direct comparison with the original PreFLMR retriever.

4.1.1 OKVQA

The OKVQA dataset comprises 9.01K questions for training and 5.05K questions for testing. The document corpus for OKVQA is derived from Wikipedia (Lin et al., 2024a), containing over 115K documents on common objects and concepts that include any of the potential answers in the OKVQA dataset. As this corpus is not officially provided with OKVQA, there are no ground truth labels for document relevance to a given query. Following Lin et al. (2024b), we use pseudo labels for training, where a document is considered a positive match with a given query if it contains any of the ground truth answers. The same document corpus is used for both training and testing.

4.1.2 EVQA

The EVQA dataset consists of 167K questions for training, 9.85K questions for validation, and 3.75K questions for testing. The M2KR implementation of EVQA is a reduced version of the original EVQA dataset. The original dataset, containing over 1 million training questions, has been condensed to remove duplicated questions that refer to the same Wikipedia entity, as well as questions requiring multiple knowledge bases. The document corpus used for EVQA is derived from WikiWeb2M (Burns et al., 2023). Unlike OKVQA, EVQA provides ground truth annotations that determine whether a document is considered a positive match for a query, serving as labels for training.

4.1.3 Difficulty of OKVQA vs EVQA

Figure 4.1 displays a few examples of OKVQA and EVQA to highlight their differences. Following Lin et al. (2024b), we take examples from their original papers to ensure an unbiased representation. In general, OKVQA is easier as questions are based on everyday concepts, such as understanding that people attend church on Sundays (first row, first image) or recognizing that fire trucks use fire hydrants (first row, second image). In contrast, EVQA requires a more specialized understanding of niche subjects, such as answering a question about "Acacia paradoxa" (second row, first image). The difficulty in VQA translates to retrieval as the model similarly needs to understand the context of the query when assessing document relevance.

However, an important consideration is also the quality of ground truth labels. OKVQA uses pseudo-labels based on whether the candidate document contains the answer to the query. This approach can lead to noisy labels that may not always be contextually relevant to the query.

In some cases, pseudo-labels are adequate. For instance, for the question about the fire hydrant (first row, second image), the pseudo-matched document with the term "fire truck" is contextually relevant:

> *Stanford Fire Truck House: The Fire Truck House, built by Charles Hodges in 1904, served as Stanford University's firehouse until the early 1970s, when the current firehouse was ...*

However, this can be problematic in other cases. For the question about 1950s style (first row, third image), the pseudo-matched document contains the term "50s", but in the context of electrical sockets:

23-50 S 10, CEI 23-50 S 11, CEI 23-50 S 16, CEI 23-50 S 17 and also NEMA 1-15 (US/Japan) plugs (older versions also had extra holes to accept UK shaver plugs). This Soviet plug, ...

In contrast, EVQA ground truth labels are manually annotated, meaning that relevant documents are always contextually relevant to the query. For instance, the relevant document for "Acacia paradoxa" (second row, first image) is a Wikipedia article about it:

The large shrub or tree up to 2 to 4 meters (7 to 13 ft) tall and has a similar width, it has ribbed branchlets that are often arched downward. It is dense with foliage; the leaves are ...

Hence, while EVQA may inherently be more challenging than OKVQA, noisy pseudo-labels can make OKVQA a more difficult dataset for experimentation. However, in practice, increasing model size and extending training steps result in improved performance metrics on OKVQA (Lin et al., 2024a,b), demonstrating that OKVQA remains suitable for evaluating model performance.

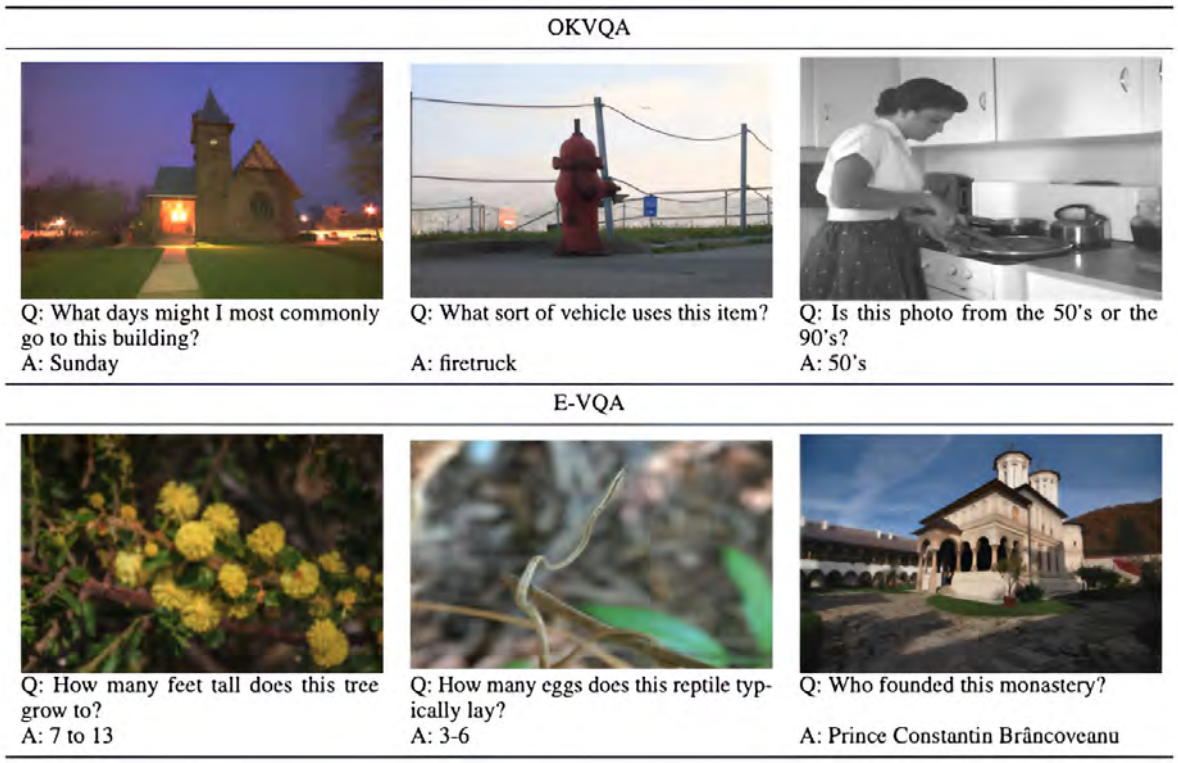


Fig. 4.1 **Comparison of OKVQA and EVQA:** OKVQA examples feature more general and everyday concepts, while EVQA examples involve more specialized and niche topics (Lin et al., 2024b)

4.2 Training Configuration

We train rerankers across all queries in the training split with a single A100 GPU. For each query, we randomly sample 1 positive document and 4 negative documents.

	monoPreFLMR | ModPreFLMR	**monoBLIP-2**
Learning Rate	10^{-5}	10^{-4}
Batch Size (Queries)	8	2
Gradient Accumulation	Every 8 Steps	Every 16 Steps
Effective Step Size (Queries)	*320*	*160*

Table 4.1 **Training Configurations:** We employ the AdamW optimizer for all models with an epsilon value of 10^{-8}, but with varying batch sizes, learning rates, and gradient accumulation strategies. Notably, monoBLIP-2 has a smaller batch size because it exceeds GPU memory capacity.

The number of cross-encoder layers, F_R is a consideration for monoPreFLMR and ModPreFLMR. For monoPreFLMR, we use a single cross-encoder layer. For ModPreFLMR, we use either 3 BERT layers or 5 IB layers ensuring that both variants have comparable parameters. For monoPreFLMR, we follow Lin et al. (2024b) and freeze the vision models during training. For monoBLIP-2, we train with LoRA (Hu et al., 2021) using an r of 8, an α of 32, and a dropout of 0.1, which substantially reduces the number of trainable parameters to only 0.1%.

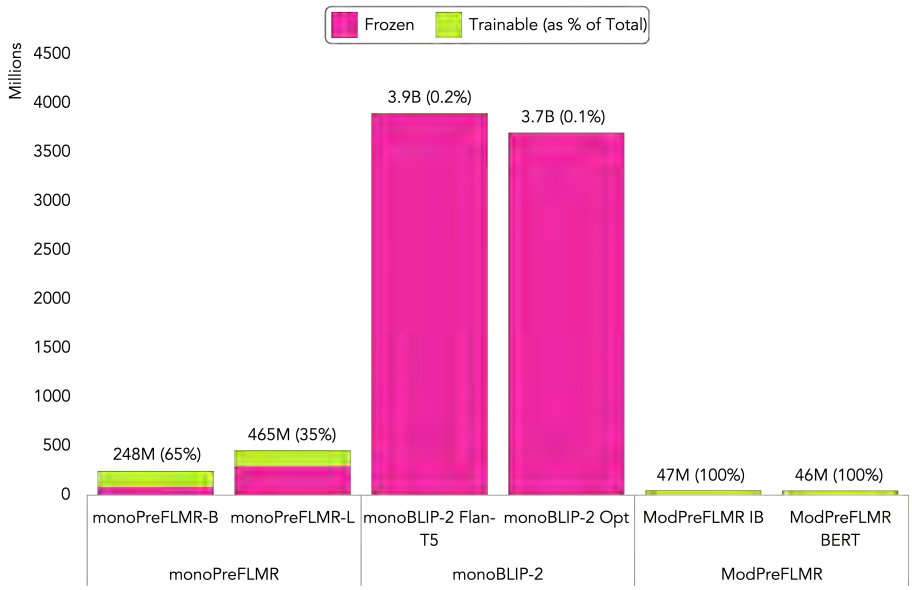


Fig. 4.2 **Reranker Parameter Sizes**: monoBLIP-2 has the most parameters, followed by monoPreFLMR and then ModPreFLMR. Despite its size, monoBLIP-2 has the fewest trainable parameters as it is fine-tuned with LoRA.

4.3 Evaluation Metrics

4.3.1 Recall

We use recall at various positions, K to evaluate performance. For an individual query, i, Recall@K$_i$ is defined as whether the correct document d_i^* exists in the top K documents of the reranked list, R_i^K (Equation 4.1).

$$\text{Recall@K}_i = \begin{cases} 1, & \text{if for query } i\text{, the correct document } d_i^* \in R_i^K \\ 0, & \text{otherwise} \end{cases} \tag{4.1}$$

Recall@K is the proportion of queries for which the correct document is found within the top K of the reranked list (Equation 4.2).

$$\text{Recall@K} = \frac{1}{N}\sum_{i=1}^{N} \text{Recall@K}_i \tag{4.2}$$

We evaluate Recall@5 as the primary metric using the test split.[9] PreFLMR is used to perform the first-stage retrieval and the top D documents are reranked. By default, we use PreFLMR-B retrieval results and set the Retrieval List Size, D, to be 100.

It is important to note that evaluation is limited to Recall@K where $K \leq D$, as we only have access to the top D documents. The performance of a reranker at Recall@D is always equal to the Recall@D of the original retriever, as reordering cannot change a binary metric of whether a document exists. Furthermore, the upper bound of reranking performance at any Recall@K is capped at the Recall@D of the original retriever as reranking cannot introduce new documents that were not already present in the retrieved list.

We use a single A100 GPU for evaluation. However, due to limited time and computation, we evaluate the first 1,040 questions in the test split. Consequently, our raw retrieval results differ from those reported by Lin et al. (2024b).

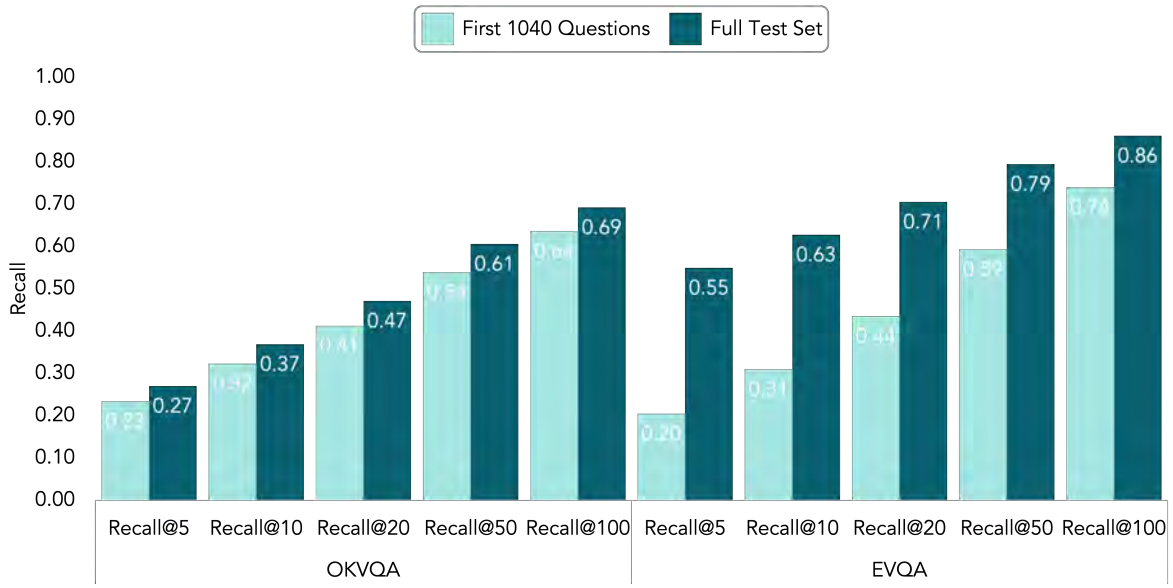


Fig. 4.3 **PreFLMR-B Recall**: The first 1,040 questions in the test split are the most challenging to retrieve, as their recall is significantly lower than the overall recall, particularly for EVQA.

4.3.2 McNemar Test

To statistically quantify the impact of a reranker, we treat the recall of each query on the test set (Equation 4.1) as a trial. Dietterich (1998) finds that the McNemar Test is the most suitable hypothesis test for comparing two models with shared trials, focusing on discordant pairs of predictions, where one model succeeds and the other fails. Under the null hypothesis, the number of discordant pairs b and c are expected to be equal, as the two models should have the

[9] We evaluate Recall@K for $K \in \{10, 20, 50\}$ in Appendix A.

same error rates. The test statistic follows a chi-square distribution with 1 degree of freedom (Equation 4.3). Unless otherwise specified in Chapter 5, rerankers that have a better recall than raw retrieval are statistically significant.

	Raw Retrieval Correct	Raw Retrieval Incorrect
Reranker Correct	a	b
Reranker Incorrect	c	d

$$\chi^2 = \frac{(|b-c|-1)^2}{b+c} \tag{4.3}$$

4.3.3 Limitations of VQA Accuracy

Although the ultimate goal of retrieval is to improve VQA Accuracy by generating answers that match ground truth responses (Goyal et al., 2017), we do not directly assess this metric. Such an assessment would require fine-tuning a text generation model, which is not feasible due to limited time and computational resources. Moreover, reporting VQA Accuracy in our work could potentially introduce misleading results. If we were to observe improvements in recall but deterioration in VQA Accuracy, this discrepancy would likely indicate one of two issues:

1. The quality of the ground truth labels may be poor, failing to accurately represent correct answers.

2. The text generator may not be well-aligned with the retrieval process, unable to effectively utilize the improved recall.

Both of these potential issues lie beyond the scope of this thesis, which focuses specifically on improving retrieval performance.

Nonetheless, it is already well established in the literature that improvements in OKVQA and EVQA retrieval recall lead to enhancements in VQA Accuracy. For OKVQA, Lin et al. (2024b) have demonstrated that an improvement in pseudo recall has a strong correlation with an improvement in VQA Accuracy. Similarly, for EVQA, Mensink et al. (2023) have shown through an oracle experiment that achieving perfect recall on the ground truth labels leads to significantly better VQA performance. Since generation models have a limited context window and only use the top retrieved documents for answer generation, Recall@5 serves as a suitable proxy metric for VQA accuracy.

4.4 Conclusion

This chapter outlines our experimental framework for investigating the effectiveness of reranking on improving document retrieval for VQA. We work with OKVQA and EVQA, both of which have ground truth labels for training. EVQA is more challenging, focused on long-tail knowledge, but includes cleaner, manually annotated labels. In contrast, OKVQA lacks ground truth labels and relies on pseudo-answer matching. We evaluate three types of rerankers: monoPreFLMR, monoBLIP-2, and ModPreFLMR using Recall@K, which measures how often the correct document appears in the top K reranked results. While we do not train a text-generation model for measuring VQA Accuracy, our approach is grounded in empirical evidence that shows that improved recall of ground truth labels correlates with better VQA Accuracy. The next chapter details experiment results.

Chapter 5

Results

In this chapter, we present the results of reranking PreFLMR retrieval on OKVQA and EVQA[10]. We begin by discussing the performance of rerankers trained with a pointwise loss function (Equation 3.13). Following this, we examine the results of using a listwise loss function (Equation 3.28) and fine-tuning on retrieved documents. We conclude with an analysis of performance across different retrievers and provide an ablation study of various reranker configurations.

5.1 Pointwise Results

We begin by training rerankers using a pointwise loss function on OKVQA and EVQA. We train each reranker for a fixed number of steps and select the best validation recall performance. Specifically, we train monoPreFLMR for 6,000 steps, monoBLIP-2 for 3,000 steps, and ModPreFLMR for 19,000 steps. We evaluate after 1,000 steps, and then every 1,000 steps afterwards for monoPreFLMR and monoBLIP-2 and every 2,000 steps for ModPreFLMR.

5.1.1 OKVQA Results

Figure 5.1 shows that monoBLIP-2 Opt consistently outperforms all other rerankers. It achieves the highest Recall@5 of 39.80% in only 2,000 steps, significantly surpassing the retrieval baseline of 23.46%. monoPreFLMR-B also shows statistically significant improvement over the retrieval baseline with a Recall@5 of 28.37% after 3,000 steps. monoPreFLMR-B performs

[10]For brevity, we present only Recall@5 as graphs in this section. It is worth noting that as we set D to 100, the upper bound of any Recall@K score is equal to and converges to the Recall@100 score of PreFLMR-B retrieval. For OKVQA, it is 63.6%, and for EVQA, it is 74.3%. For more detailed results in tabular form, please refer to Appendix A.

marginally better than monoPreFLMR-L, indicating that a larger vision encoder is not necessary. Although not statistically significant, ModPreFLMR IB also shows improvement over the retrieval baseline with a Recall@5 of 25.19% after 11,000 steps, the same as ModPreFLMR BERT. The superior recall of monoBLIP-2 Opt comes at the computational cost of having the slowest inference time of 12.64 seconds per question.[11] However, in practice, this can be faster as reranking is parallelizable; the scores for query-document pairs are independent and can be allocated to multiple GPUs. While there is a trade-off in performance, ModPreFLMR IB has only 47M parameters and the fastest inference time of 0.13 seconds per query, which is 97x faster than monoBLIP-2 Opt. Despite achieving the same recall, the ModPreFLMR IB is twice as fast as ModPreFLMR BERT, as attention is performed only on the query (Section 3.4.1).

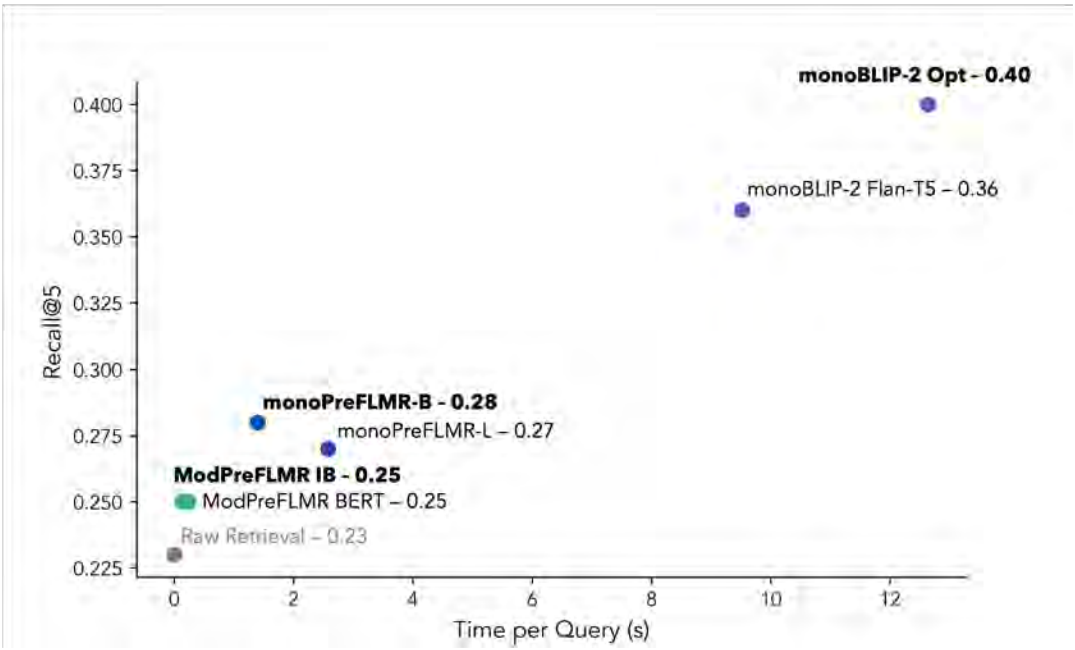


Fig. 5.1 **OKVQA Recall@5 vs Efficiency (Pointwise Loss):** There is a clear trade-off between performance and efficiency. monoBLIP-2 Opt achieves the highest recall but has the longest processing time. In contrast, monoPreFLMR-B and ModPreFLMR offer quicker responses but with more modest recall.

5.1.2 EVQA Results

Figure 5.2 shows similar trends when reranking EVQA, although less relative improvement over retrieval as it is more challenging. monoBLIP-2 Flan-T5 outperforms other rerankers, achieving the highest Recall@5 of 30.48% in only 1,000 steps, compared to the retrieval baseline of 20.48%. However, monoPreFLMR demonstrates much lower performance than

[11] Measured on average, with near identical times across different evaluations.

the retrieval baseline. After 6,000 steps, monoPreFLMR-B only achieves a Recall@5 of 7.5%, which is much lower than the retrieval baseline. Again, monoPreFLMR-B trains much faster than monoPreFLMR-L. As a bonus experiment, we continued training monoPreFLMR-B using a pointwise loss function up to 15,000 steps, improving Recall@5 to 9.9% and showing that there is room for improvement with more training steps. For EVQA, ModPreFLMR BERT is not only faster, but also offers substantially better performance than monoPreFLMR-B, with a Recall@5 of 17.59% after 15,000 steps, however, it still falls short of raw retrieval, let alone monoBLIP-2 Flan-T5.

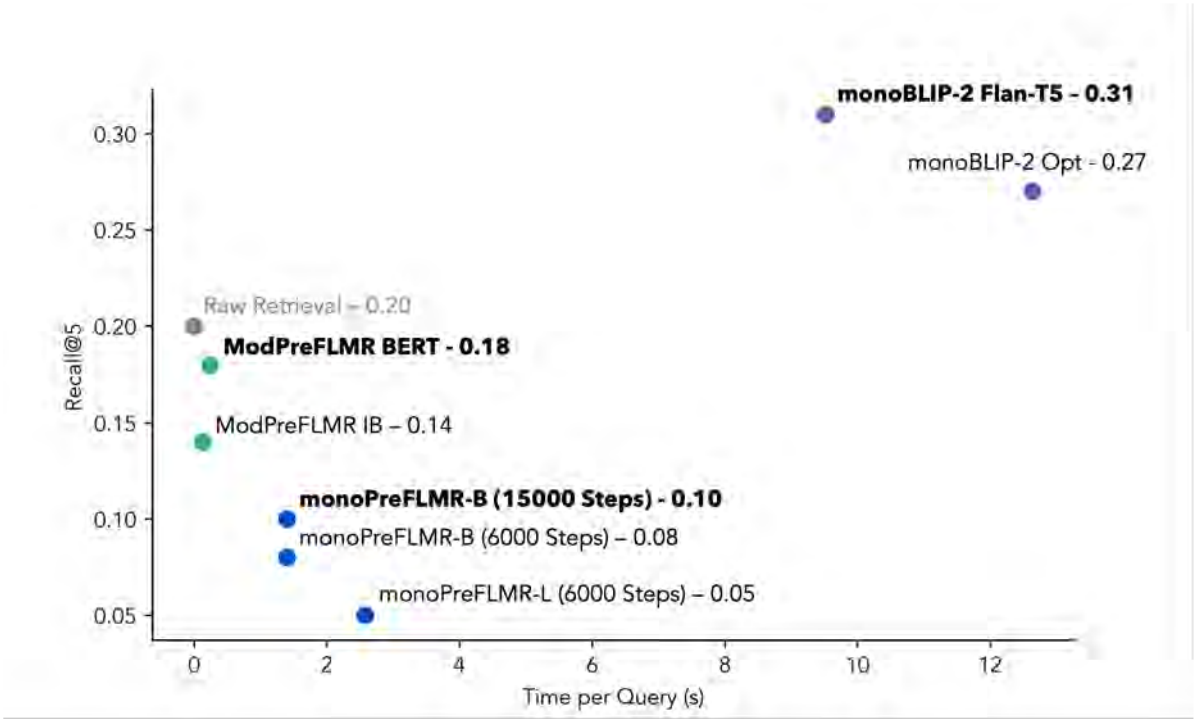


Fig. 5.2 **EVQA Recall@5 vs Efficiency (Pointwise Loss):** With the more challenging EVQA, monoBLIP-2 is the only reranker that improves performance over raw retrieval. Additionally, ModPreFLMR demonstrates both superior efficiency and performance compared to monoPreFLMR.

5.1.3 Impact of Pre-Training

The performance of monoBLIP-2 is supported by Brown et al. (2020) who demonstrate that sequence-to-sequence models are excellent zero-shot learners. Scaling laws have shown that larger models are significantly more sample-efficient, and the best method of training is on a modest amount of data and stopping before convergence (Kaplan et al., 2020). In our case, monoBLIP-2 has 10x more parameters than monoPreFLMR and achieves the best performance in just 1,000 steps before overfitting on EVQA (Figure 5.3). This is in contrast

to monoPreFLMR, which continues to learn even after 15,000 steps without yet beating raw retrieval. The success of sequence-to-sequence models over encoder models in text reranking is corroborated by findings from Nogueira et al. (2020), which show that monoBERT performs poorly without many training examples, sometimes worse than BM25 raw retrieval, whereas monoT5 performs significantly better with fewer training examples.

While the larger parameter size likely plays a role, it is possible that the source of performance stems from the sequence-to-sequence models having more extensive pre-training tasks. Nogueira et al. (2020) has shown that even a monoT5 model with fewer parameters than monoBERT yields better results with less training data. However, as the smallest checkpoint for BLIP-2 is over 3B parameters compared to the largest monoPreFLMR-L with 465M parameters, a like-for-like comparison was not explored in this thesis. The advantage of sequence-to-sequence models in low-resource settings may instead be attributed to their ability to generate rather than classify, allowing them to learn from more complex generation tasks. Nogueira et al. (2020) suggests that the decoder contains pre-trained information on the meaning of tokens. They observed that choosing more contextually relevant options like "yes/no" versus "apple/orange" significantly impacts performance when there is little training data.

The importance of leveraging pre-training information is further demonstrated in the effectiveness of ModPreFLMR compared to monoPreFLMR. Figure 5.3 shows that while this is not evident in OKVQA, a more basic task, it becomes more apparent in EVQA, where ModPreFLMR demonstrates a significant advantage in training speed over the monoPreFLMR. This is because using the embeddings from the PreFLMR retrieval stage as inputs essentially leverages the knowledge of the PreFLMR retrieval model in measuring relevance between the query and document. While it could be argued that monoPreFLMR is also based on the pre-trained PreFLMR, it only uses the query encoder and applies it out of context to both the query and document. Consequently, it does not match the pre-trained task as well as the ModPreFLMR where the document is passed separately through the document encoder. This would explain the slow training speed, as it is possible that we are training the query encoder of monoPreFLMR more or less from scratch.

Indeed, the performance disparities in Figure 5.3 across different reranker types are quite exaggerated when compared to results from Nogueira et al. (2020), where at convergence, the differences between monoBERT and monoT5 are much smaller. It is possible that more training steps are required before actual convergence, whereas we may have stopped ours too early at a local plateau with limited steps. Another possible reason for this discrepancy is that our training dataset may be too small, as we are only using OKVQA and EVQA in isolation. We could follow the example of Lin et al. (2024b) and pre-train across more datasets such as LLaVA, OVEN, WIT, CC3M, KVQA, and Infoseek.

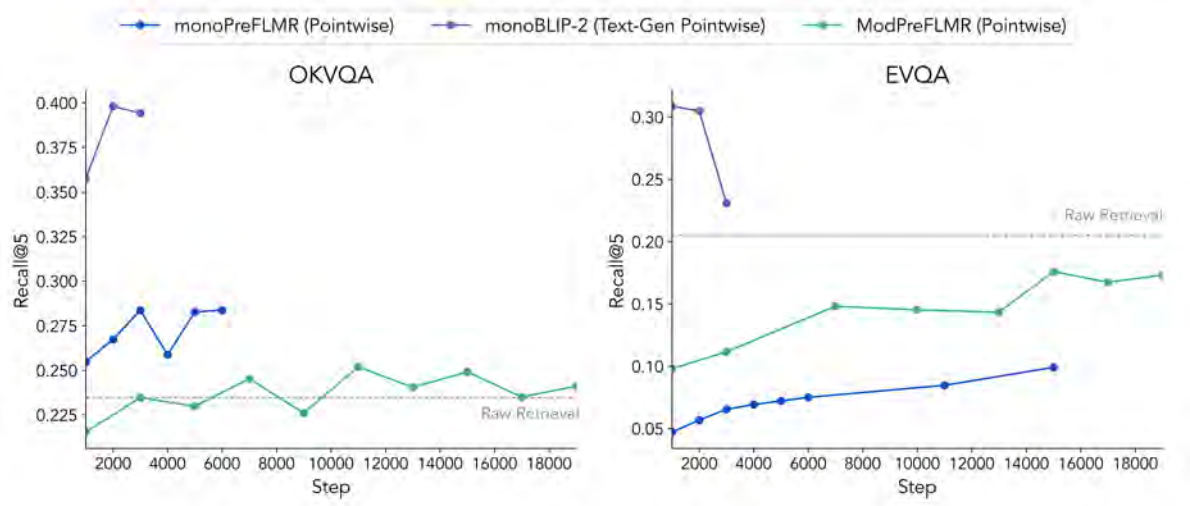


Fig. 5.3 **Recall@5 vs Training Step (Pointwise Loss):** monoBLIP-2 reaches the highest performance at a much faster pace than monoPreFLMR and ModPreFLMR, highlighting the impact of extensive pre-training on model performance.

5.1.4 Pointwise Loss Versus Recall

The pointwise loss operates on individual query-document scores and does not fully align with ranking multiple documents together. As an illustrative example, we examine the validation pointwise loss of the ModPreFLMR IB model on OKVQA. We average the cross-entropy loss for all 100 query-document pairs across the 1,040 queries in the test set, totaling 104,000 individual losses. We then compare this to Recall@5 when ranking with these scores. Figure 5.4 shows that the validation loss does not correspond to recall with the relationship appearing somewhat arbitrary.

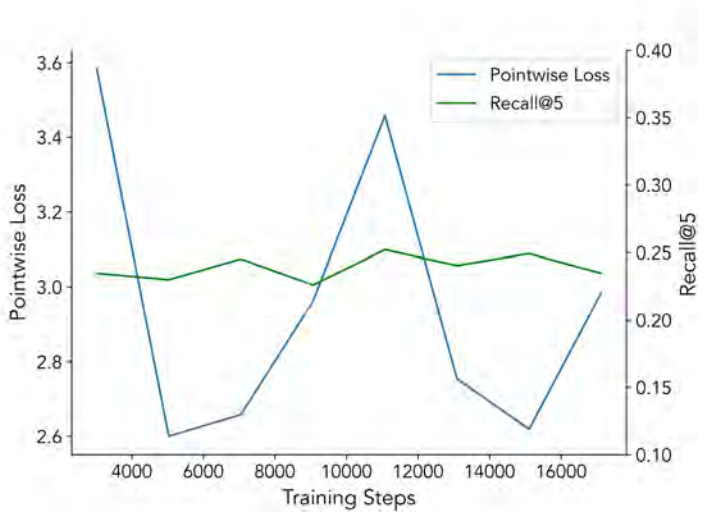


Fig. 5.4 **Pointwise Loss vs Recall (OKVQA ModPreFLMR IB):** There is no clear relationship between pointwise loss and recall during validation.

Figure 5.5 shows that the reason for this is because the average loss of a correctly ranked query (whether a correct document is in the top 5) is the same as the average loss of an incorrectly ranked query, and hence the loss does not provide any information about recall.

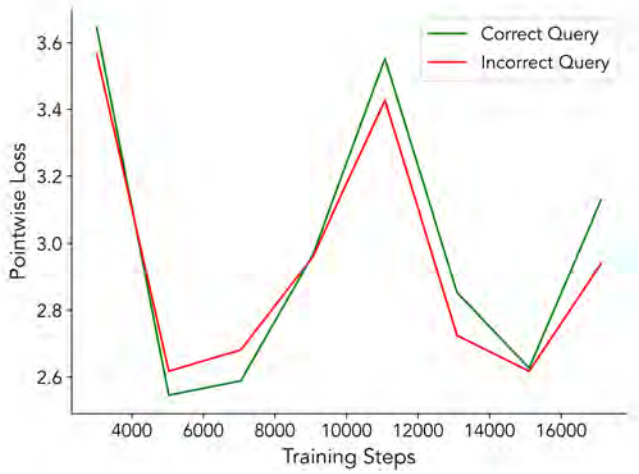


Fig. 5.5 **Pointwise Loss vs Training Step (OKVQA ModPreFLMR IB):** The average loss of correctly ranked queries and incorrectly ranked queries are nearly identical.

Figure 5.6 shows that the root cause of this issue comes from the magnitude of the predicted logit, $z_{q,d}$. When the model predicts larger logits, the loss is almost always higher, regardless of whether the query is ranked correctly (the slope of the red and green trend lines are similar). This is because the validation set is imbalanced, with 2,456 positive and 101,544 negative

samples. A model that predicts lower logits is considerably more likely to have a lower cross-entropy loss and the average magnitude of the scores becomes the key driver of the loss rather than their relative rankings.

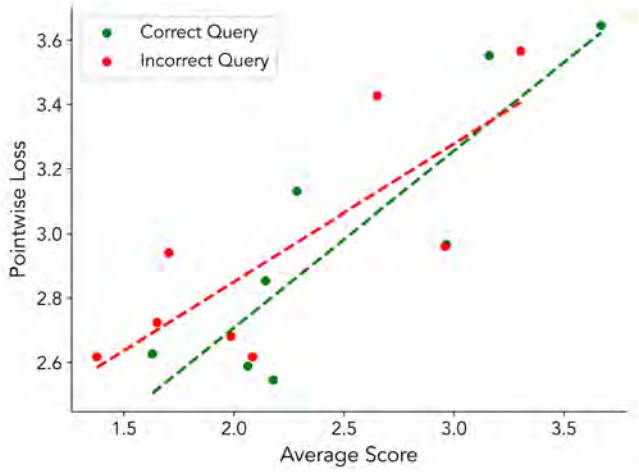


Fig. 5.6 **Pointwise Loss vs Logit (OKVQA ModPreFLMR IB):** Due to class imbalance, higher logits correspond to higher losses, regardless of the correctness of the ranking.

An imbalanced dataset is inherently characteristic of ranking. This problem parallels classification where a good cross-entropy loss is not always correlated with accuracy, particularly with imbalanced datasets on minority classes where their accuracy is overshadowed by the majority class (He and Garcia, 2009). In classification, we consider an instance to be positive if its probability exceeds a defined threshold and similarly in ranking for Recall@K, we define this threshold to be the document with the K^{th} highest probability. Consequently, the idea of validating using cross-entropy loss becomes problematic, and it is more important to analyze the performance on recall directly, rather than focusing on the pointwise loss.

5.2 Listwise Loss Function

We showcase the results of using a listwise loss function, implemented through the listwise softmax, on training performance. These models are trained from the pointwise checkpoint of the best-performing model type (bolded in Figure 5.1 and Figure 5.2), as the only difference between listwise and pointwise training is the loss function. Both methods perform training and inference using the output of a single logit. We train an additional 4,000 steps for monoPreFLMR, 2,000 steps for monoBLIP-2, and 8,000 steps for ModPreFLMR, evaluating every 2,000 steps. Unlike other models which have more or less stabilised (Figure 5.3), evaluating

monoPreFLMR for EVQA further is not productive at this stage, as the model trained with a pointwise loss function is no where near convergence yet, making a fair comparison impractical.

Figure 5.7 demonstrates that training with a listwise loss yields performance comparable to, if not slightly worse than, the pointwise loss across all rerankers. As discussed in Section 3.5.3, the listwise gradient (Equation 3.30) is effective at differentiating between highly similar and confounding samples within a batch. However, when sampling noisy negatives from the entire corpus, it is not difficult to differentiate between positive and negative samples. A model trained with listwise loss may not generalize effectively to inference time, where retrieved documents representing the top candidates would be more difficult to differentiate. On the other hand, the pointwise gradient (Equation 3.29) continues to push scores towards the ground truth during training, regardless of whether it is easily outperforming the noisy negatives, which translates to better performance at inference time when faced with more challenging distinctions between documents.

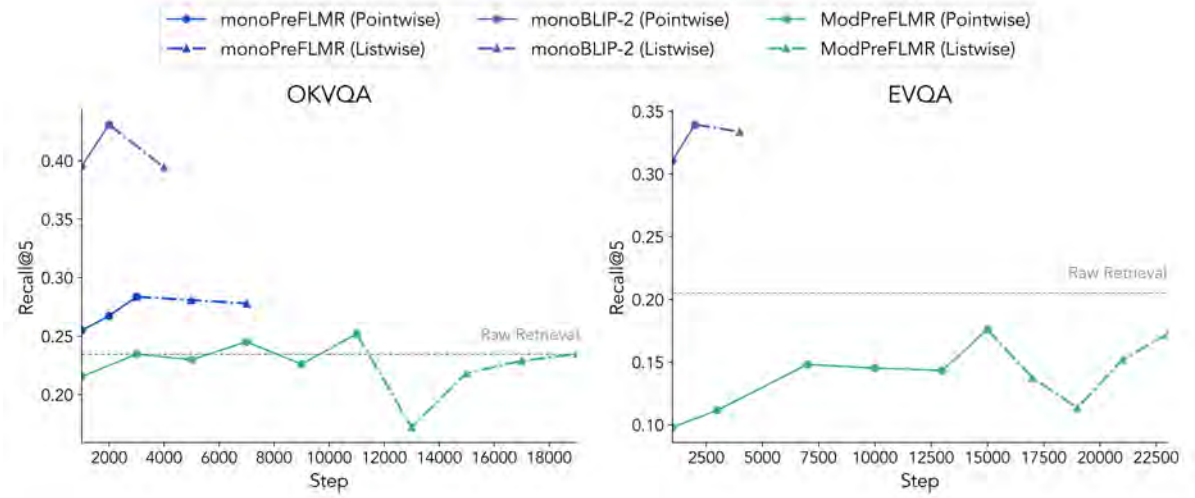


Fig. 5.7 **Recall@5 vs Training Step (Listwise Loss):** When sampling noisy negatives from the full corpus, the listwise loss yields slightly worse performance than the pointwise loss.

5.2.1 Listwise Loss Versus Recall

The listwise loss operates across multiple document scores and aligns more closely with the ranking task than the pointwise loss. We examine the average validation loss of the listwise ModPreFLMR IB model on OKVQA. To calculate the listwise loss for a given query, we randomly sample one positive document and four negative documents from the retrieved list 500 times and take the average. We skip queries where the retrieved list does not contain any positive documents, resulting in an evaluation on 662 of the 1040 queries. We then compare

this to Recall@5 when ranking with these scores. Figure 5.8 shows that, while the validation loss still does not correspond perfectly to recall, it is much less noisy than the pointwise loss.

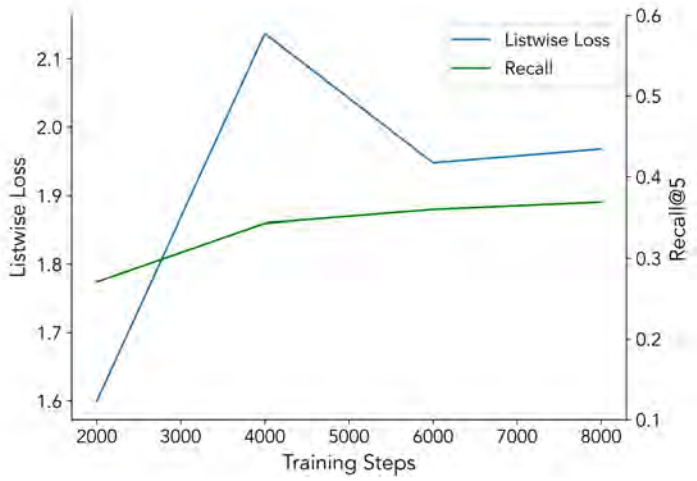


Fig. 5.8 **Listwise Loss vs Recall (OKVQA ModPreFLMR IB):** The listwise loss does not perfectly correlate with recall performance, though it fluctuates much less than the pointwise loss.

The source of improvement is evident in Figure 5.9, which shows that the listwise loss for an incorrect query is now clearly higher than for a correct query, whereas previously there is no noticeable difference for the pointwise loss.

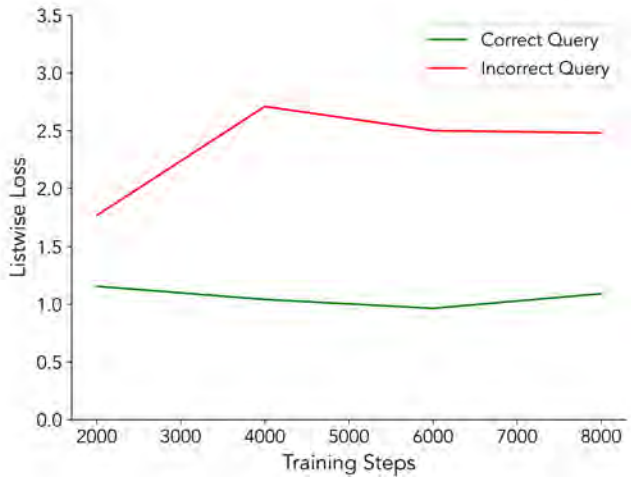


Fig. 5.9 **Listwise Loss vs Training Step (OKVQA ModPreFLMR IB):** There is now a clear difference between the average loss of correctly ranked queries and incorrectly ranked queries.

Figure 5.10 shows that the magnitude of the score, $r_{q,d}$ has a much smaller impact on loss. However, it remains imperfect for incorrect queries where larger scores can result in larger losses when the listwise loss measures the difference in magnitude between the positive document's score and the negative documents' scores (Equation 3.28). Although the listwise loss is much closer to recall than the pointwise loss, validating remains noisy, making it crucial to analyze performance based on recall directly.

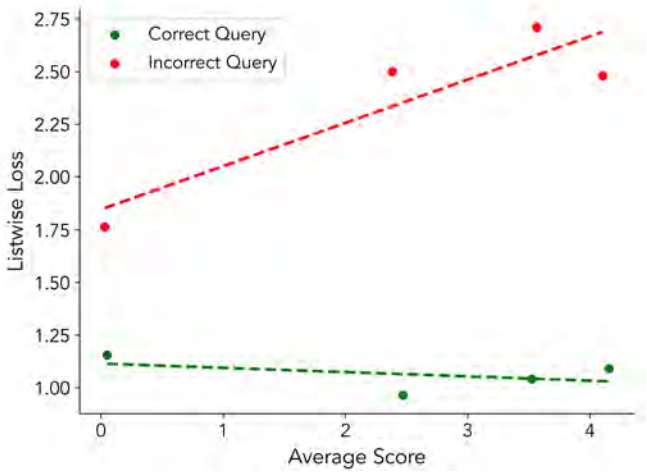


Fig. 5.10 **Listwise Loss vs Logit (OKVQA ModPreFLMR IB):** The value of the scores has a smaller impact on listwise loss than on pointwise loss, although significant enough to make it an unreliable indicator of recall.

5.2.2 Direct Prediction for Sequence-to-Sequence Rerankers

We experiment with training monoBLIP-2 using a pointwise loss on the output of a single logit (Equation 3.32) instead of a text generation loss. Losing text generation is akin to losing pre-trained information on the meaning of tokens (Section 5.1.3). However, Nogueira et al. (2020) found that when rerankers have been trained with sufficient steps, as in our case, this is no longer important. In fact, Figure 5.11 shows that training with a single logit leads to even better performance than text generation, suggesting that predicting with a single score is more effective than consolidating the outputs of two different scores.[12] These findings corroborate with Section 5.5.1, which also shows that a sigmoid function outperforms a softmax function for monoPreFLMR.

[12]It is possible to begin training from the checkpoint of the best text generation model. However, we find that training with a single logit already outperforms text generation performance in just a few steps, making this unnecessary.

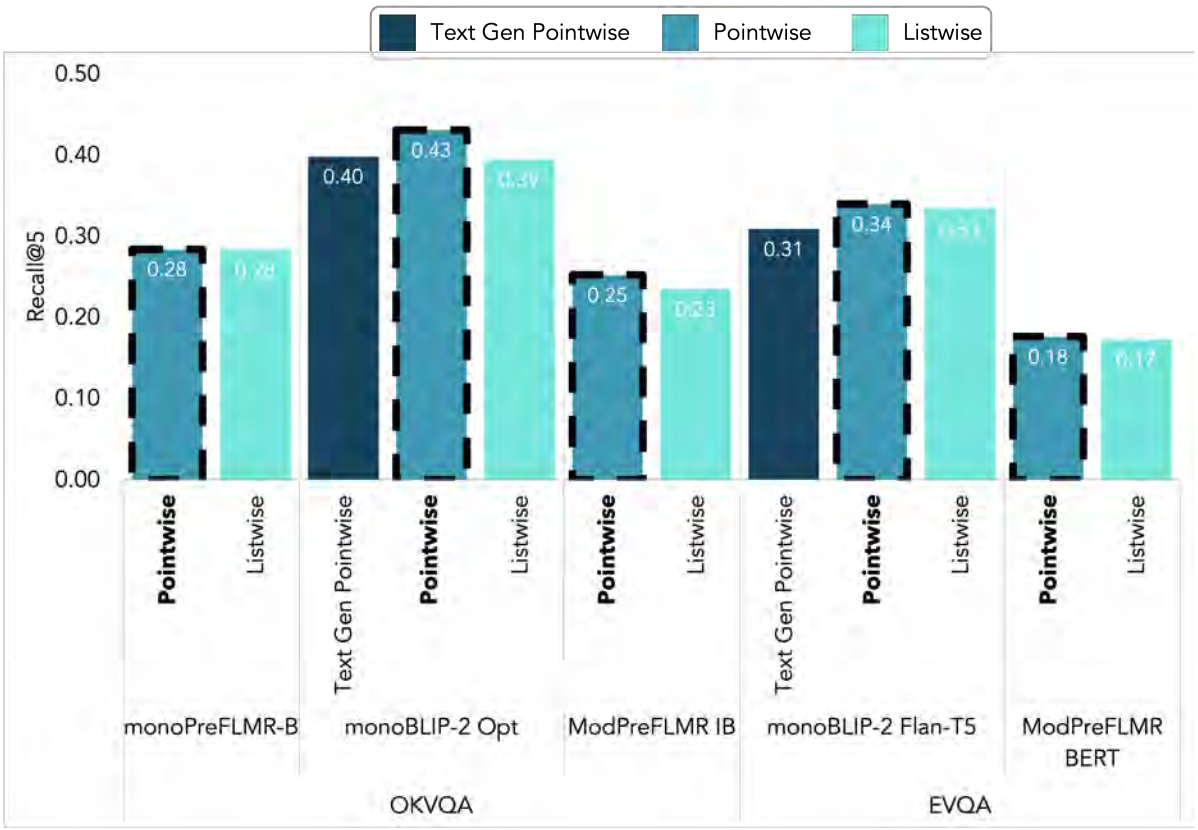


Fig. 5.11 **Recall@5 vs Loss Functions** (*the highlighted bar represents the best loss for the respective model*): The listwise loss performs slightly worse than the pointwise loss when training on noisy negatives sampled from the full corpus. Furthermore, for monoBLIP-2, training the pointwise loss with a single output consistently leads to better performance than using a text generation output.

5.3 Fine-Tuning on Retrieved Documents

We investigate fine-tuning on documents to understand the distribution of first-stage PreFLMR-B retrieval. We experiment with various sampling methods, and for each method, we begin from the previous best checkpoint and continue training: 4,000 steps for monoPreFLMR, 2,000 steps for monoBLIP-2, and 8,000 steps for ModPreFLMR, evaluating performance every 2,000 steps.

5.3.1 Fine-Tuning on Raw Retrieval

We explore random sampling from retrieval results to allow rerankers to learn the imbalance towards negatives (Section 5.1.4). As there are no architectural changes, we initialize from the best-performing pointwise model checkpoints (bolded in Figure 5.1 and Figure 5.2).

As intended, Figure 5.12 shows that the validation loss decreases significantly across all datasets and rerankers, however, there is varied improvements in recall. In general, fine-tuning on retrieved documents leads to significantly improved recall, the most substantial being ModPreFLMR on OKVQA which achieves a Recall@5 of 27.8% and now achieves statistically significant performance over raw retrieval. Moreover, monoPreFLMR on OKVQA improves to a recall of 32.6%, and monoBLIP-2 on EVQA improves to a Recall@5 of 36.4%. Trends in Figure 5.13 suggest that these models could achieve even better performance with more steps.

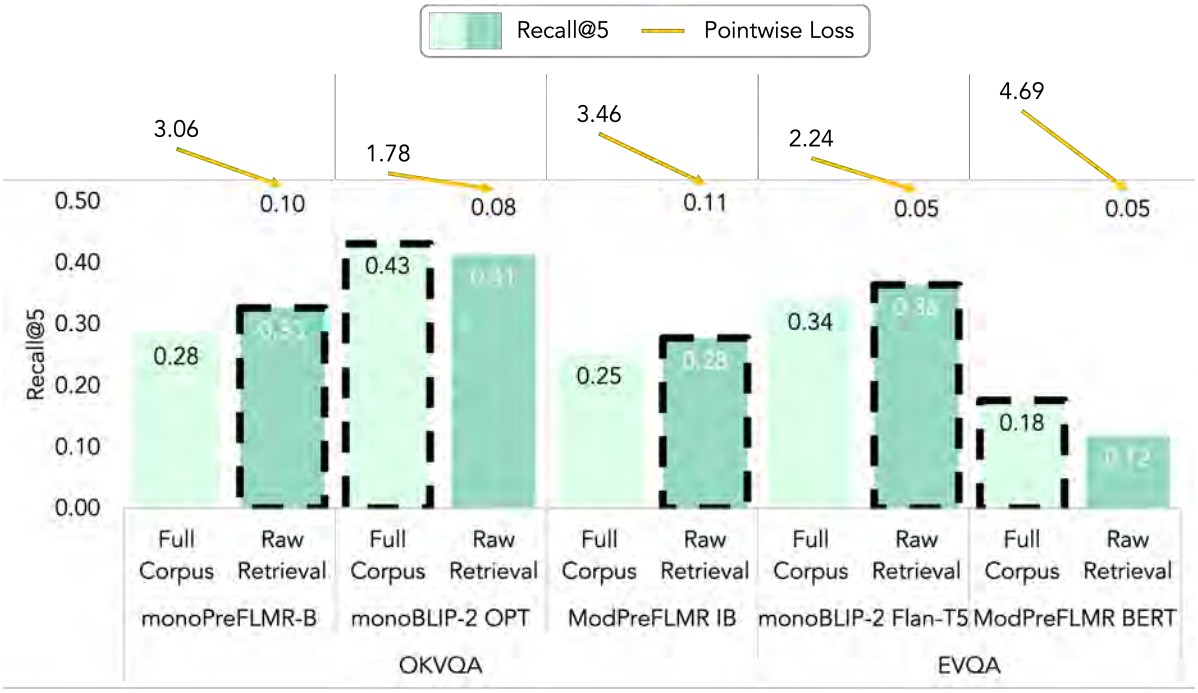


Fig. 5.12 **Fine-tuning on Full Corpus vs Raw Retrieval** (*the highlighted bar represents the best sampling method for the respective model*): Despite a significant decrease in validation loss when fine-tuning on retrieved documents, improvements in recall vary among models.

However, we observe a significant drop in performance for ModPreFLMR on EVQA. This can be attributed to its substantial post-retrieval class imbalance, where only 0.74% of documents are positive, compared to OKVQA where 2.3% of documents are positive. Extensive research has shown that imbalanced datasets lead to suboptimal accuracy on the minority class, with the model learning to predict consistently low logits to minimize the loss function (He and

Garcia, 2009). In this case, with an overwhelming number of negative documents, the model learns to predict negatives and loses the information from the positives, as it is an easier task to learn to predict negative than to learn the relative difference between positive and negative. Consequently, while the ModPreFLMR works well for OKVQA, its performance on EVQA is not as strong. monoBLIP-2 for EVQA may not suffer as severely from imbalance as it has much fewer trainable parameters, benefiting from learning the distribution of retrieval without being as susceptible to overfitting.

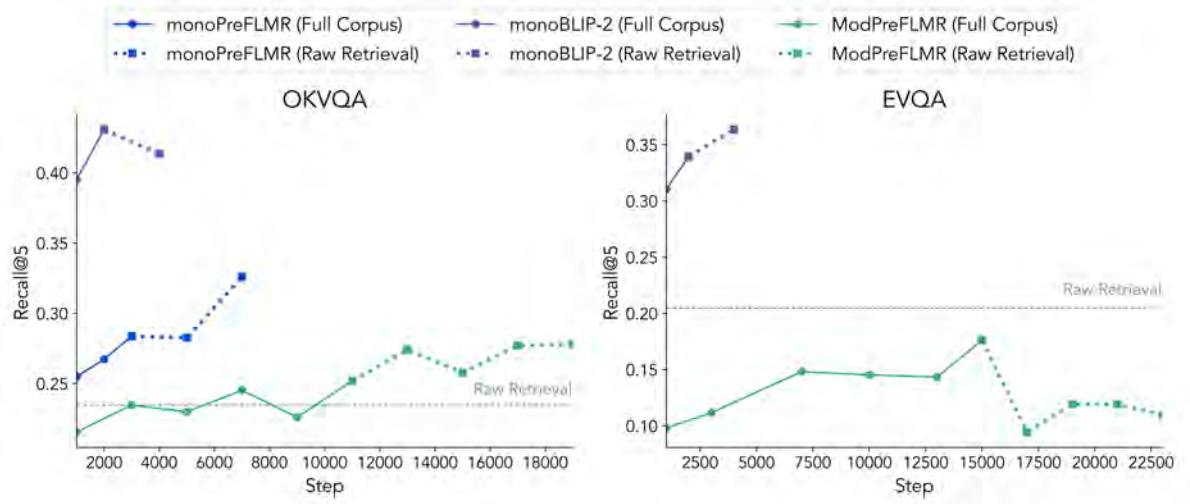


Fig. 5.13 **Recall@5 vs Training Step (Fine-Tuning on Raw Retrieval):** In general, there are performance improvements when fine-tuning on retrieved documents. However, for ModPreFLMR on EVQA, performance drops dramatically due to class imbalance.

5.3.2 Fine-Tuning on Upsampled Retrieval

To address imbalance in retrieval results, we upsample positive samples, similar to when training on the full corpus. In this case, we sample from first-stage PreFLMR-B retrieval, similarly selecting 1 positive document and 4 negative documents. As there is no guarantee a positive document exists in the retrieved results, if one does not exist, we sample a random document from the full corpus. This modification effectively addresses class imbalance while ensuring the model continues to learn from documents representative of the retrieval results.

We initialize from the checkpoint of the best-performing pointwise models (bolded in Figure 5.1 and Figure 5.2). Figure 5.14 shows improved performance for ModPreFLMR on EVQA, which achieves a 21.1% recall, above raw retrieval, although, it is not statistically significant. Moreover, monoBLIP-2 on EVQA demonstrates the best performance yet, with a recall of 38.3%.

However, this approach introduces a new challenge with OKVQA as the fine-tuning process becomes significantly less stable across all rerankers. In some instances, we find that the training algorithm collapses and begins to overfit with training losses approaching zero and leading to a substantial drop in recall during validation. This instability can be attributed to both the significantly smaller size of OKVQA (which is approximately 95% smaller than EVQA) and the lack of diversity in top retrieval results, given that it is an easier task and based on pseudo-labelling (Section 4.1.3). This means that OKVQA has a high probability of a similar, if not the same, positive document sampled in successive batches. A pointwise loss means that the same update is likely to be performed repeatedly in a similar direction as $s_{q,d}$ is calculated in isolation (Equation 3.29). This can potentially lead to unstable training as the model overfits and reinforces noise from the same documents rather than learn the true decision boundary. Indeed, this is not as much of an issue with a larger dataset like EVQA, which offers more diversity in samples. Consequently, the probability that confounding documents are sampled successively is lower, and the reranker does not overfit on it.

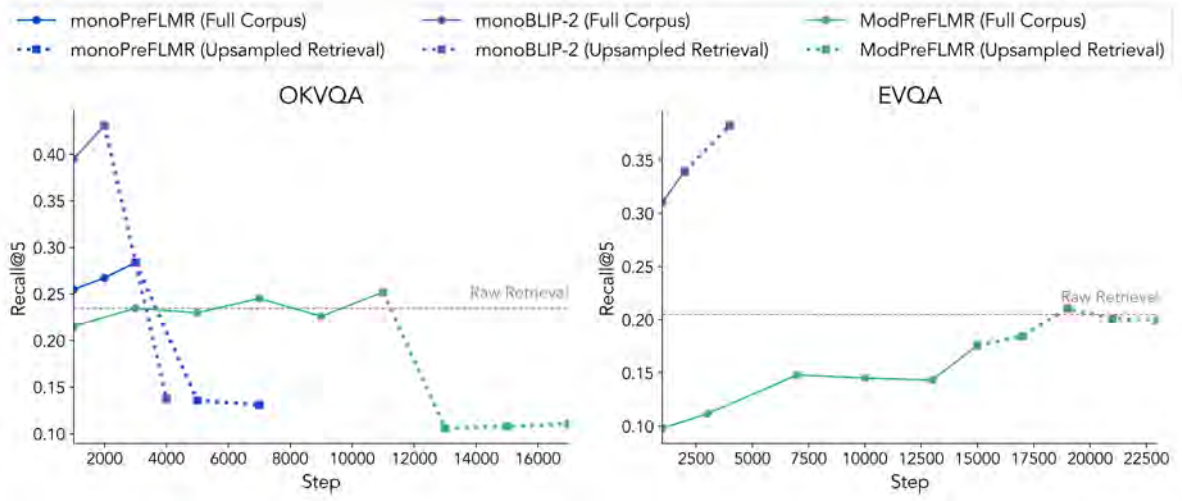


Fig. 5.14 **Recall@5 vs Training Step (Fine-Tuning on Upsampled Retrieval):** Sampling a fixed ratio of positive and negative samples when fine-tuning on retrieved documents addresses class imbalance and improves recall for EVQA. However, this significantly reduces the diversity of training samples, causing instability in OKVQA, which is a smaller dataset.

5.3.3 Fine-Tuning on Upsampled Retrieval with Listwise Loss

To address training instability when using a pointwise loss, we revisit the listwise loss. We find that another benefit of the listwise approach is its robustness to overfitting from oversampling, which is a significant issue with pointwise loss on small datasets. Instead, with a listwise loss, $s_{q,d}$ is not a constant between batches as the score is influenced by the negatives in the batch

(Equation 3.30). Even if the same positive document is sampled again, the negatives in the batch will not be the same, so its score, $s_{q,d}$, and hence the associated gradient, will not be similar. This ensures that the gradient does not flow excessively in a similar direction, thereby avoiding overfitting on the same sample. In effect, the other documents in the batch act as a regularizer by adding diversity to the sample despite sampling the same positive document.

With a listwise loss, we initialize from the checkpoint of the best-performing listwise models (shown in Figure 5.11). Figure 5.15 shows that training is more stable for OKVQA, leading to significant improvements in recall across all rerankers, including monoBLIP-2. The listwise approach has addressed both imbalance issues and challenges in upsampling small datasets. However, it is worth noting that for EVQA, the listwise loss does not offer a substantial advantage over the pointwise loss, and upsampling is sufficient. This is possibly because EVQA is a more difficult task with more niche documents, and even at the top of the retrieval list, there is diversity between documents. Similar to sampling noisy negatives across the full corpus, the listwise loss is not needed to differentiate scores between documents and instead, it is better for a pointwise loss to focus on pushing scores towards ground truth labels.

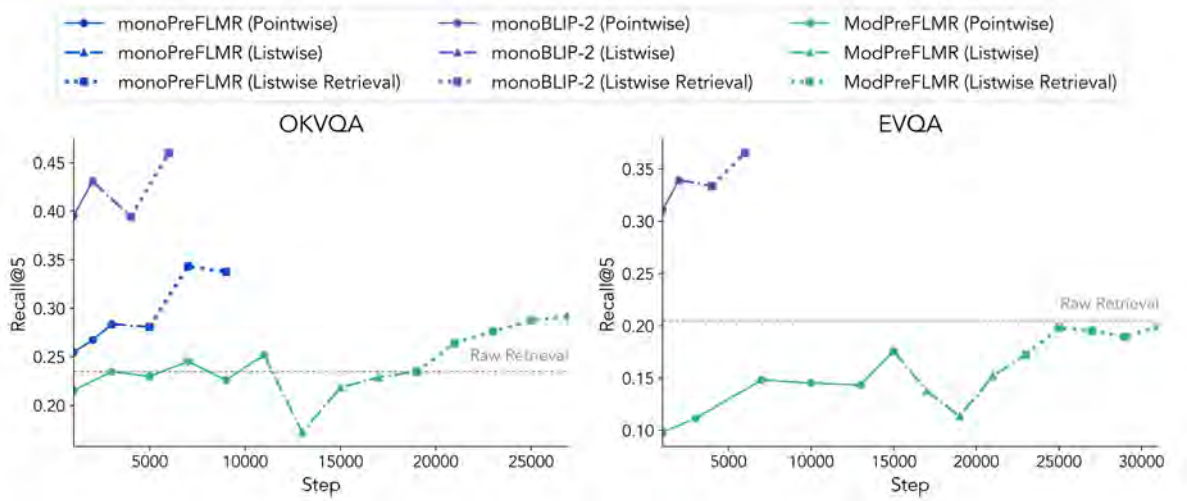


Fig. 5.15 **Recall@5 vs Training Step (Fine-Tuning on Upsampled Retrieval with Listwise Loss):** Listwise loss is beneficial when there are highly confounding documents among the retrieved documents in small datasets like OKVQA, as it stabilizes training and improves performance.

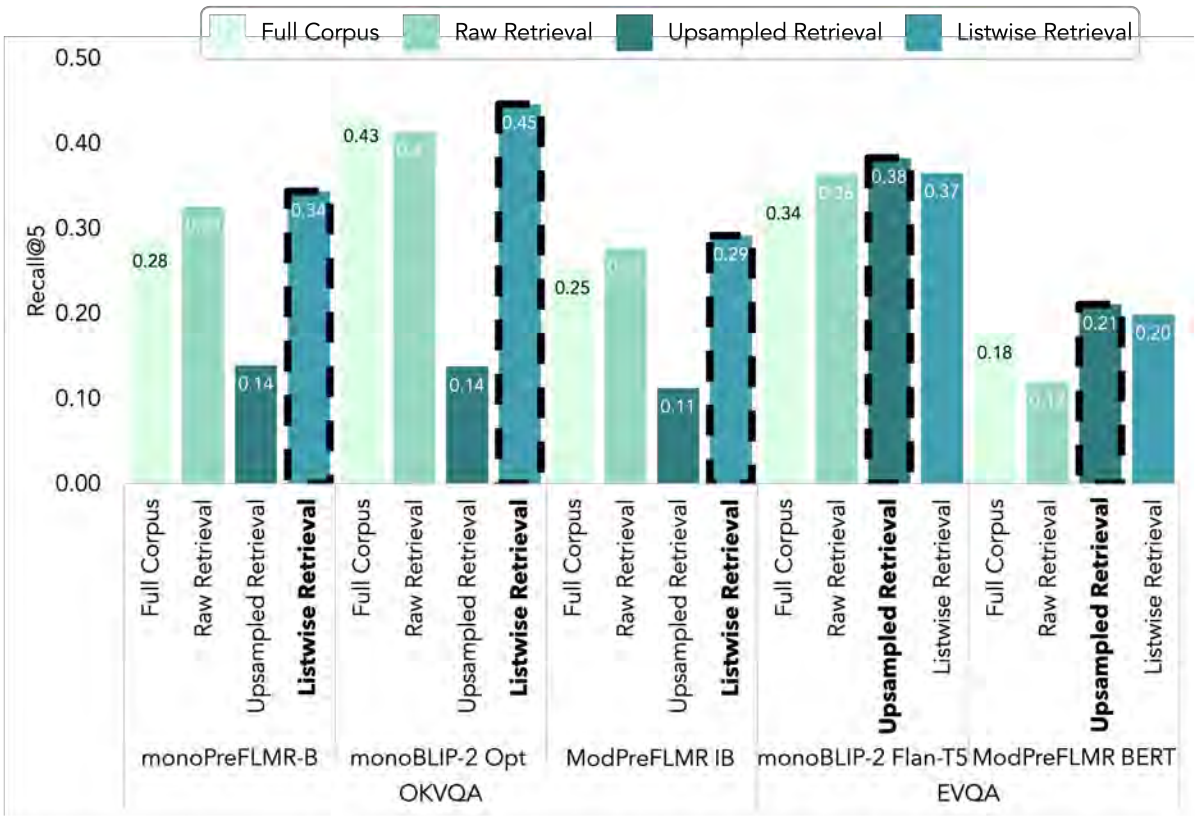


Fig. 5.16 **Recall@5 vs Sampling Method** (*the highlighted bar represents the best sampling method for the respective model*): For all rerankers, fine-tuning on retrieved documents leads to improvements in performance. However, it is important to ensure a balanced ratio of positive to negative documents when sampling. For OKVQA, it is better to use the listwise loss as there are highly confounding documents, whereas for EVQA with more diverse samples, it is better to use the pointwise loss as ground truth labels provide more direction.

5.4 Overall Results

We conclude by consolidating results for the best rerankers for PreFLMR-B retrieval (bolded in Figure 5.16[13]) across all recall positions and also test on PreFLMR-L and PreFLMR-G retrieval results.[14] Table 5.1 demonstrates that rerankers not only improve Recall@5, but also Recall at other positions. Similarly, they can also improve the performance of the better PreFLMR-L and PreFLMR-G retrieval results. However, it is interesting to note that the post-reranking recall appears to be constant across all the retrieval results and hence, the most significant impact is

[13]For monoPreFLMR-B for EVQA which was not trained on retrieved documents, we use the pointwise version from Figure 5.2

[14]We do not test ModPreFLMR on PreFLMR-L and PreFLMR-G retrieval due to limited time and computation as they require training a new model on respective retrieval embeddings.

on the weakest PreFLMR-B retrieval result. It is possible that this is because the reranker was fine-tuned on PreFLMR-B retrieval results and is most effective when inference is under that distribution Gao et al. (2021).

Data	Retriever	Reranker	Recall@5	Recall@10	Recall@20	Recall@50
OKVQA	PreFLMR-B	*Raw Retrieval*	*0.2346*	*0.3231*	*0.4125*	*0.5394*
		monoPreFLMR	0.3432	0.4326	0.5000	0.5875
		monoBLIP-2	0.4461	0.5067	0.5653	0.6201
		ModPreFLMR	0.2913	0.3855	0.4615	0.5711
	PreFLMR-L	*Raw Retrieval*	*0.2740*	*0.3721*	*0.4529*	*0.5904*
		monoPreFLMR	0.3269	0.4250	0.5105	0.6096
		monoBLIP-2	0.4596	0.5355	0.5836	0.6451
	PreFLMR-G	*Raw Retrieval*	*0.2750*	*0.3683*	*0.4654*	*0.6038*
		monoPreFLMR	0.3451	0.4480	0.5355	0.6370
		monoBLIP-2	0.4750	0.5423	0.6000	0.6673
EVQA	PreFLMR-B	*Raw Retrieval*	*0.2048*	*0.3105*	*0.4355*	*0.5932*
		monoPreFLMR	0.0990	0.1548	0.2615	0.5442
		monoBLIP-2	0.3826	0.4936	0.5865	0.7057
		ModPreFLMR	0.2105	0.2961	0.4336	0.5769
	PreFLMR-L	*Raw Retrieval*	*0.3528*	*0.4394*	*0.5682*	*0.7307*
		monoPreFLMR	0.1134	0.1932	0.3557	0.6413
		monoBLIP-2	0.3961	0.5086	0.6163	0.7605
	PreFLMR-G	*Raw Retrieval*	*0.3269*	*0.4326*	*0.5548*	*0.7240*
		monoPreFLMR	0.1230	0.2019	0.3442	0.6538
		monoBLIP-2	0.3990	0.5134	0.6336	0.7740

Table 5.1 **Overall Results**: Rerankers that improve Recall@5 for PreFLMR-B retrieval can also consistently improve results across other recall positions and retrieval results.

5.5 Ablation Studies

We present ablation studies on various reranking configurations and compare them against the default settings (Chapter 4) used in the main results.

5.5.1 Sigmoid Activation

We compare measuring probabilities with two scores and softmax activation versus a single score and sigmoid activation. The original monoBERT model uses a softmax and the theoretical benefit is that two scores adds stability by addressing the saturation of the sigmoid at tails. However, using a sigmoid removes half the redundant classification layer parameters with a single output, while retaining the constraint that the output must be less than 1. We conducted an early experiment to determine the appropriate activation function, training monoPreFLMR-B on OKVQA for 1,000 steps. Figure 5.17 shows that the model performs faster when predicting a single score with a sigmoid activation. This aligns with Section 5.2, where training on a single logit is superior to training on all logits for monoBLIP-2.

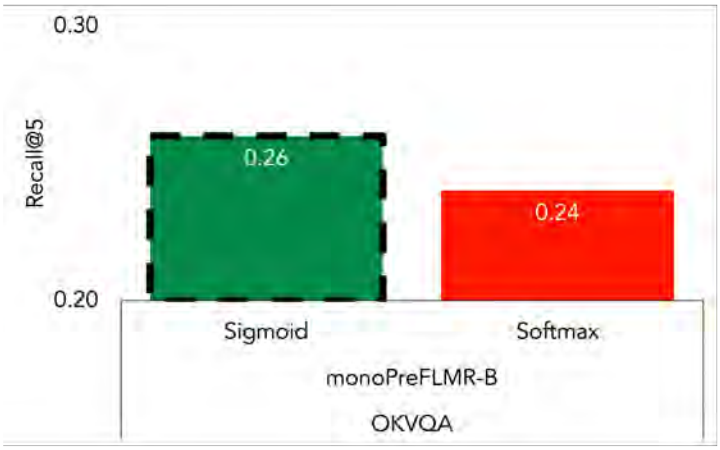


Fig. 5.17 **Sigmoid Activation Ablation:** Predicting using a single score with sigmoid activation improves training speed by reducing redundant parameters, compared to using two scores with softmax activation.

5.5.2 Encoder Vision Model

We conduct ablation studies on completely removing the vision model, motivated by monoPreFLMR-B's superior performance over monoPreFLMR-L on both OKVQA and EVQA, despite having a smaller vision model. Removing the vision model makes monoPreFLMR equivalent to monoBERT. Figure 5.18 shows that for OKVQA, no vision model yields results better than monoPreFLMR-L but marginally worse than monoPreFLMR-B, reaffirming that image information is not as important when reranking. However, for EVQA, removing the vision model worsens results compared to monoPreFLMR-L, indicating visual features are important for this challenging task, reaffirming findings in Lin et al. (2024b) that a better vision model matters more for EVQA than OKVQA.

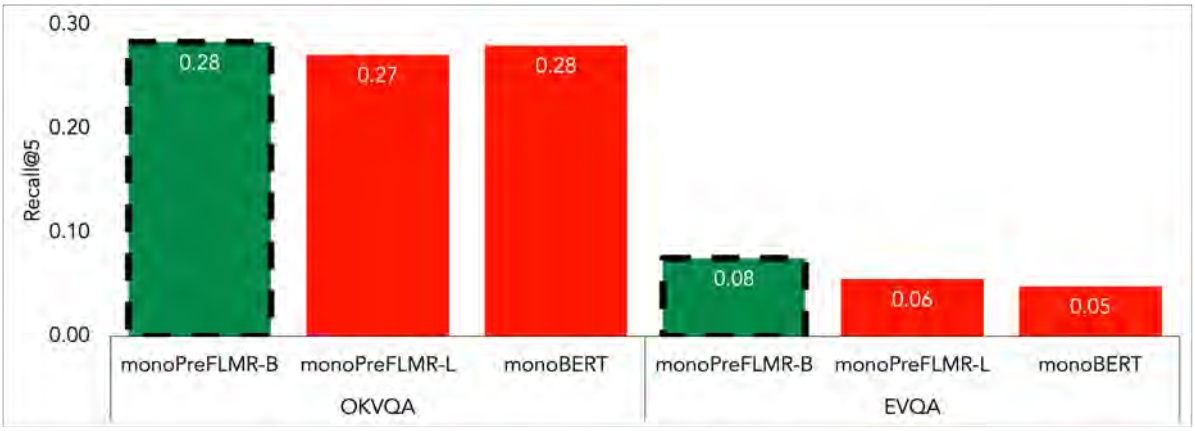


Fig. 5.18 **Encoder Vision Model Ablation:** Removing the vision model worsens performance, although it is much worse on EVQA than OKVQA, highlighting the differing importance of visual features between these tasks.

5.5.3 Cross-Encoder Layers

We conduct ablation studies on the number of cross-encoder layers, F_R, for monoPreFLMR and ModPreFLMR. Additional layers theoretically allow better information processing through deeper token interactions. This is particularly relevant for monoPreFLMR, where text embeddings $Q_{q,d}$ only attend to image embeddings $\left[Q_I^{MLP} \mid Q_I^{TR}\right]$ in the cross-encoder (Equation 3.15). Thus, a deeper cross-encoder might better incorporate image information and improve performance. However, Figure 5.19 shows that for monoPreFLMR, a single transformer block suffices to summarize information and adding cross-encoder layers yield diminishing returns, suggesting they may require longer training or lead to overfitting. Similarly, 5 IB layers are sufficient for ModPreFLMR IB, aligning with Gao et al. (2020), who demonstrate that excessive IB blocks increase computational cost without improving performance.

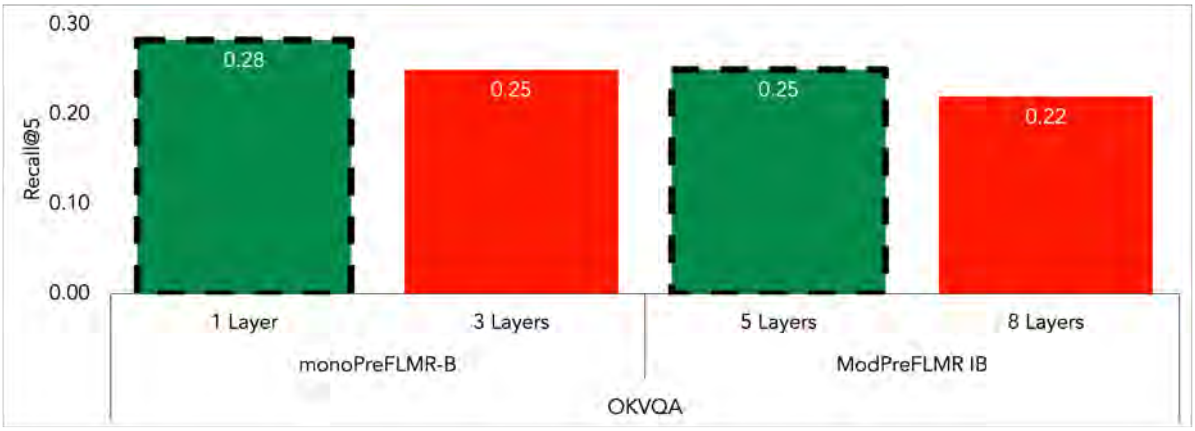


Fig. 5.19 **Cross-Encoder Layers Ablation:** A single cross-encoder layer for monoPreFLMR and 5 IB layers for ModPreFLMR are sufficient, with additional layers providing diminishing returns.

5.5.4 Freezing Vision Model

By default, we follow PreFLMR retrieval by freezing vision encoders. As an ablation study, Figure 5.20 confirms that training the vision model of monoPreFLMR decreases performance. This suggests that pre-trained vision encoders contain sufficient information, aligning with general insights that overfitting is a concern when training rerankers. The most successful models are those that are well pre-trained with fewer trainable parameters during fine-tuning, such as monoBLIP-2 with LoRA and ModPreFLMR, which uses retrieval embeddings followed by a small cross-encoder.

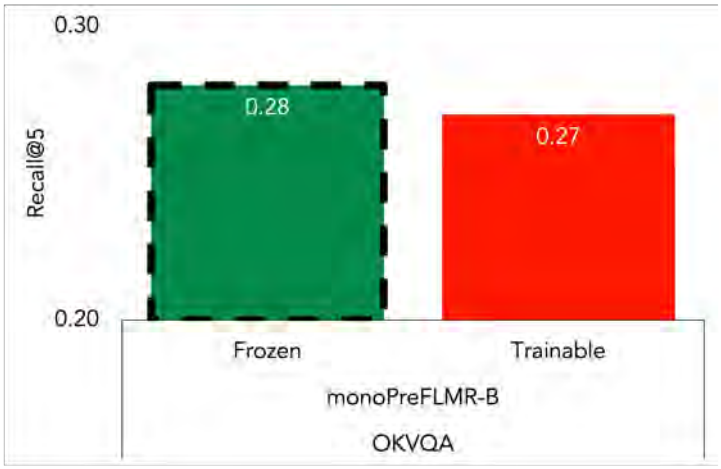


Fig. 5.20 **Freezing Vision Model Ablation:** Freezing the vision model in monoPreFLMR improves performance, confirming that it is important to avoid overfitting.

5.5.5 Retrieval List Size

We investigate the effect of varying the number of documents from PreFLMR-B retrieval, D, on Recall@5. When $D = 5$, the result is equivalent to the raw retrieval outcome, as reranking the top 5 documents does not affect Recall@5.

Figure 5.21 reveals that the more powerful the reranker, the better it is to have a larger set of documents for reranking. Notably, for monoBLIP-2, setting $D = 100$ yields the best performance on both OKVQA and EVQA datasets, with performance continuously decreasing with fewer documents. Theoretically, a larger D allows the more powerful reranker to process more documents, potentially improving accuracy but at the cost of decreased efficiency.

However, this trend does not hold for less powerful rerankers where setting D too high can inadvertently introduce lower quality documents into the top rankings. For instance, with monoPreFLMR and ModPreFLMR on OKVQA, the optimal is $D = 75$ and $D = 50$ while for ModPreFLMR on EVQA, the optimal is $D = 10$. Any value below these thresholds provides insufficient documents for effective ranking, while higher values introduce excessive noise. However, in extreme cases, such as monoPreFLMR on the EVQA dataset, the reranker fails to outperform raw retrieval at all values of D and there is a continuous decrease in recall as we allow it to rank more documents.

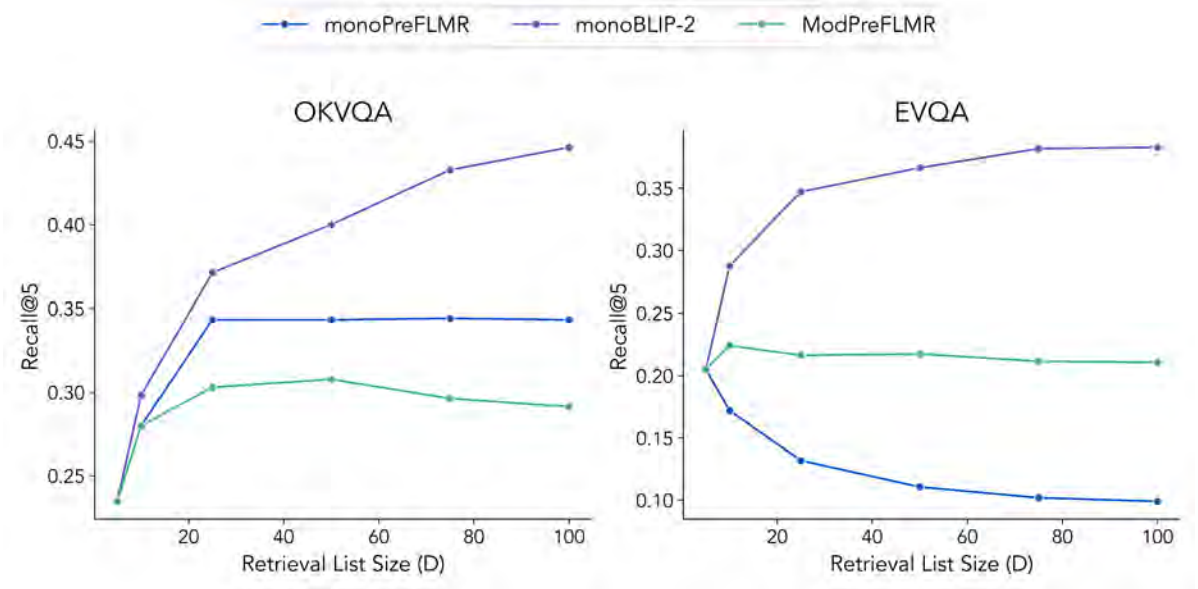


Fig. 5.21 **Recall@5 vs Retrieval List Size:** The performance of rerankers improves with more documents for powerful models like monoBLIP-2, while less powerful models perform optimally with less document due to introduced noise.

5.6 Conclusion

In general:

- For **OKVQA, all rerankers achieve statistically significant improvements over PreFLMR-B retrieval**. For **EVQA, monoBLIP-2 shows statistically significant improvements**, while monoPreFLMR and ModPreFLMR do not.
- **monoBLIP-2 achieves the best performance with the fewest training steps**, aligning with Nogueira et al. (2020) who suggest sequence-to-sequence models excel at few-shot learning.
- **ModPreFLMR is 97x faster than monoBLIP-2 while still significantly improving recall over PreFLMR**. This introduces the **feasibility of mid-interaction mechanisms** as a **middle ground in the performance-efficiency trade-off** between late-interaction PreFLMR and early-interaction monoBLIP-2.
- **Rerankers also improve recall for more powerful PreFLMR-L and PreFLMR-G retrievers**, though the impact is more pronounced on PreFLMR-B. Gao et al. (2021) suggest that performance could be further improved by fine-tuning on respective retrievers to better align document distributions.

In terms of training:

- **Training on retrieved documents significantly improves recall compared to training on the entire corpus**, as it better reflects the distribution encountered during inference. However, it is important to sample a balanced ratio of positive to negative documents to address class imbalance.
- **A listwise loss is better in smaller datasets like OKVQA, where top-end documents are more homogeneous**, as it helps distinguish between documents and stabilizes training. **Conversely, a pointwise loss is better for larger datasets like EVQA, where top-end documents are still diverse**, as ground truth labels can provide more information than the other documents in the batch.

In terms of reranker configurations:

- **Reranking benefits significantly from pre-trained knowledge but is prone to overfitting.** The best-performing models, monoBLIP-2 and ModPreFLMR, have few trainable parameters but extensive pre-training. Ablation studies show that using a single output with sigmoid activation is sufficient. A small vision encoder that is frozen is also sufficient, as is a small cross-encoder to consolidate pre-trained information.

- **While reranking typically balances speed and accuracy, this trade-off holds primarily for well-trained powerful rerankers like monoBLIP-2**, where more documents lead to better recall. **For weaker rerankers, there is an optimal number of documents to provide;** exceeding this can introduce noise and degrade performance.

In summary, our results suggest that an effective approach to training rerankers is to **start with well-pretrained models**, align them on the reranking task using a **pointwise loss across diverse datasets**, and then **fine-tune them with a listwise loss on specific retrievers**.

Chapter 6

Conclusion

This thesis has presented a comprehensive exploration of reranking techniques to improve document retrieval for KB-VQA. Despite significant advancements in VLMs, the challenge of answering questions that require external knowledge remains substantial. Our work addresses this gap by extending reranking methods from text retrieval to the multimodal domain, specifically targeting the enhancement of retrieval systems for VQA tasks.

We proposed three novel reranking architectures: monoPreFLMR, monoBLIP-2, and ModPreFLMR. Our rerankers demonstrate statistically significant improvements over the state-of-the-art PreFLMR retriever across OKVQA and EVQA. The monoBLIP-2 model shows the best performance in few-shot learning scenarios, aligning with literature suggesting that sequence-to-sequence models train faster due to better pre-training. We also implement a modularized approach with ModPreFLMR, which initializes with embeddings from the PreFLMR retrieval stage in what we introduce as a mid-interaction mechanism. This mechanism employs a smaller cross-encoder that achieves effective performance over late-interaction mechanisms like PreFLMR while maintaining significantly lower computational overhead compared to early-interaction mechanisms like monoBLIP-2.

Our experiments confirmed that training on samples of retrieved documents, rather than the entire corpus, leads to better generalization and performance across all rerankers and datasets. We also validate the benefits of training with a listwise loss, which is particularly effective for stabilizing training and improving recall for datasets like OKVQA where retrieved documents are more homogeneous. However, we found that when documents are more diverse, as in the case of the more complex EVQA, the pointwise loss performs better by separating individual losses. We then conducted various ablation studies with results that reinforce the importance of preventing overfitting and leveraging information from pre-training. From these results, we draw the conclusion that training the optimal reranker involves leveraging strong pre-trained

models, training them on diverse reranking datasets with a pointwise loss, and then fine-tuning on specific downstream retrievers with a listwise loss.

6.1 Limitations

We acknowledge some limitations of this work from imposed time constraints of 4 months and computational resources of 1000 GPU hours. As reported, we only train rerankers for a limited number of steps as the purpose of this thesis is to explore the dynamics of training rather than to completion. We spend our GPU hours training multiple configurations to understand what works best. Each configuration takes a significant amount of time to train and evaluate. Indeed, trends in recall show that some rerankers could benefit from additional training steps or more diverse data to further improve performance meaning means that some of our results are likely to be understated.

In regards to evaluation, we only report performance on the first 1,040 queries of the test set due to computational limitations. For reference, monoBLIP-2 requires 12.4 seconds to rerank per query. Including overhead, it takes around 8 GPU hours to evaluate 1,040 queries for a single checkpoint. The OKVQA test set, with over 5,000 queries, would require 40 GPU hours for full evaluation, while the EVQA test set, with over 3,000 queries, would require 24 GPU hours. Furthermore, we have chosen not to report VQA Accuracy as it is contingent on valid ground truth labels and a strong text generation model, factors that are beyond the control of this thesis. While we have ensured that this is a like-for-like comparison against reported benchmarks, a more comprehensive study could evaluate on the full test set and include VQA Accuracy.

6.2 Future Work

Beyond training rerankers to completion and evaluating further, we suggest two other avenues for future work that could be explored.

6.2.1 Pre-Training Diversity

- The slow training speed of our encoder reranker, monoPreFLMR, could stem from an insufficient pre-training task which does not align well with reranking. We could experiment with other encoders pre-trained on different tasks, such as VisualBERT (Li et al., 2019), UNITER (Chen et al., 2020), and LXMERT (Tan and Bansal, 2019).

- Training encoder rerankers is data-intensive (Nogueira et al., 2020) and our training set may be too small, particularly OKVQA which has shown evidence of overfitting. To address this, we could follow Lin et al. (2024b) and pre-train our rerankers across more diverse datasets, such as LLaVA, OVEN, WIT, CC3M, KVQA, and Infoseek, before fine-tuning and evaluating on OKVQA and EVQA.

- The benefit of sequence-to-sequence models is their extensive pre-training. However, efficiency with monoBLIP-2 may be a concern, and we could experiment with smaller models which Nogueira et al. (2020) suggest can perform just as well. This is challenging because monoBLIP-2 is already relatively small at 3B parameters, and the smallest VLMs are around 1.5-2B parameters in size, like Gemini Nano (Team et al., 2023) and MobileVLM (Chu et al., 2024). Nevertheless, this might be unnecessary, as Nogueira et al. (2020) suggests that encoder rerankers should offer comparable performance with sufficient training.

6.2.2 Knowledge Distillation

- As our rerankers improve recall on the retriever, their scores could be used for knowledge distillation. The retriever can be trained on these softer labels to learn more nuanced decision boundaries, which can be particularly helpful for harder cases (Hofstätter et al., 2020).

- Knowledge could also be distilled from the retriever to the reranker as a form of regularization, particularly when overfitting is a concern with smaller datasets. We can experiment with injecting retrieval scores as token inputs into rerankers (Askari et al., 2023). Moreover, for ModPreFLMR, which initializes with PreFLMR retrieval embeddings whose dot products represent token-level similarities, these similarities can be used as a prior in cross-encoder layers as a form of attention fusion.

References

Aishwarya Agrawal, Dhruv Batra, Devi Parikh, and Aniruddha Kembhavi. Don't just assume; look and answer: Overcoming priors for visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 4971–4980, 2018.

Stanislaw Antol, Aishwarya Agrawal, Jiasen Lu, Margaret Mitchell, Dhruv Batra, C Lawrence Zitnick, and Devi Parikh. Vqa: Visual question answering. In *Proceedings of the IEEE international conference on computer vision*, pages 2425–2433, 2015.

Arian Askari, Amin Abolghasemi, Gabriella Pasi, Wessel Kraaij, and Suzan Verberne. Injecting the bm25 score as text improves bert-based re-rankers. In *European Conference on Information Retrieval*, pages 66–83. Springer, 2023.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*, 2016.

Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

Andrea Burns, Krishna Srinivasan, Joshua Ainslie, Geoff Brown, Bryan A Plummer, Kate Saenko, Jianmo Ni, and Mandy Guo. Wikiweb2m: A page-level multimodal wikipedia dataset. *arXiv preprint arXiv:2305.05432*, 2023.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*, 2017.

Sirui Chen, Yuan Wang, Zijing Wen, Zhiyu Li, Changshuo Zhang, Xiao Zhang, Quan Lin, Cheng Zhu, and Jun Xu. Controllable multi-objective re-ranking with policy hypernetworks. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 3855–3864, 2023a.

Xi Chen, Xiao Wang, Soravit Changpinyo, AJ Piergiovanni, Piotr Padlewski, Daniel Salz, Sebastian Goodman, Adam Grycner, Basil Mustafa, Lucas Beyer, et al. Pali: A jointly-scaled multilingual language-image model. *arXiv preprint arXiv:2209.06794*, 2022.

Yang Chen, Hexiang Hu, Yi Luan, Haitian Sun, Soravit Changpinyo, Alan Ritter, and Ming-Wei Chang. Can pre-trained vision and language models answer visual information-seeking questions? *arXiv preprint arXiv:2302.11713*, 2023b.

Yen-Chun Chen, Linjie Li, Licheng Yu, Ahmed El Kholy, Faisal Ahmed, Zhe Gan, Yu Cheng, and Jingjing Liu. Uniter: Universal image-text representation learning. In *European conference on computer vision*, pages 104–120. Springer, 2020.

Mehdi Cherti, Romain Beaumont, Ross Wightman, Mitchell Wortsman, Gabriel Ilharco, Cade Gordon, Christoph Schuhmann, Ludwig Schmidt, and Jenia Jitsev. Reproducible scaling laws for contrastive language-image learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2818–2829, 2023.

Xiangxiang Chu, Limeng Qiao, Xinyu Zhang, Shuang Xu, Fei Wei, Yang Yang, Xiaofei Sun, Yiming Hu, Xinyang Lin, Bo Zhang, et al. Mobilevlm v2: Faster and stronger baseline for vision language model. *arXiv preprint arXiv:2402.03766*, 2024.

Hyung Won Chung, Le Hou, Shayne Longpre, Barret Zoph, Yi Tay, William Fedus, Yunxuan Li, Xuezhi Wang, Mostafa Dehghani, Siddhartha Brahma, et al. Scaling instruction-finetuned language models. *Journal of Machine Learning Research*, 25(70):1–53, 2024.

Paul Covington, Jay Adams, and Emre Sargin. Deep neural networks for youtube recommendations. In *Proceedings of the 10th ACM conference on recommender systems*, pages 191–198, 2016.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

Thomas G Dietterich. Approximate statistical tests for comparing supervised classification learning algorithms. *Neural computation*, 10(7):1895–1923, 1998.

Feng Gao, Qing Ping, Govind Thattai, Aishwarya Reganti, Ying Nian Wu, and Prem Natarajan. Transform-retrieve-generate: Natural language-centric outside-knowledge visual question answering. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5067–5077, 2022.

Luyu Gao, Zhuyun Dai, and Jamie Callan. Modularized transfomer-based ranking framework. *arXiv preprint arXiv:2004.13313*, 2020.

Luyu Gao, Zhuyun Dai, and Jamie Callan. Rethink training of bert rerankers in multi-stage retrieval pipeline. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR 2021, Virtual Event, March 28–April 1, 2021, Proceedings, Part II 43*, pages 280–286. Springer, 2021.

Yash Goyal, Tejas Khot, Douglas Summers-Stay, Dhruv Batra, and Devi Parikh. Making the v in vqa matter: Elevating the role of image understanding in visual question answering. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 6904–6913, 2017.

Liangke Gui, Borui Wang, Qiuyuan Huang, Alex Hauptmann, Yonatan Bisk, and Jianfeng Gao. Kat: A knowledge augmented transformer for vision-and-language. *arXiv preprint arXiv:2112.08614*, 2021.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Mingwei Chang. Retrieval augmented language model pre-training. In *International conference on machine learning*, pages 3929–3938. PMLR, 2020.

Shuguang Han, Xuanhui Wang, Mike Bendersky, and Marc Najork. Learning-to-rank with bert in tf-ranking. *arXiv preprint arXiv:2004.08476*, 2020.

Haibo He and Edwardo A Garcia. Learning from imbalanced data. *IEEE Transactions on knowledge and data engineering*, 21(9):1263–1284, 2009.

Sebastian Hofstätter, Sophia Althammer, Michael Schröder, Mete Sertkan, and Allan Hanbury. Improving efficient neural ranking models with cross-architecture knowledge distillation. *arXiv preprint arXiv:2010.02666*, 2020.

Edward J Hu, Yelong Shen, Phillip Wallis, Zeyuan Allen-Zhu, Yuanzhi Li, Shean Wang, Lu Wang, and Weizhu Chen. Lora: Low-rank adaptation of large language models. *arXiv preprint arXiv:2106.09685*, 2021.

Hexiang Hu, Yi Luan, Yang Chen, Urvashi Khandelwal, Mandar Joshi, Kenton Lee, Kristina Toutanova, and Ming-Wei Chang. Open-domain visual entity recognition: Towards recognizing millions of wikipedia entities. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 12065–12075, 2023a.

Ziniu Hu, Ahmet Iscen, Chen Sun, Zirui Wang, Kai-Wei Chang, Yizhou Sun, Cordelia Schmid, David A Ross, and Alireza Fathi. Reveal: Retrieval-augmented visual-language pre-training with multi-source multimodal knowledge memory. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 23369–23379, 2023b.

Drew A Hudson and Christopher D Manning. Gqa: A new dataset for real-world visual reasoning and compositional question answering. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 6700–6709, 2019.

Huiqiang Jiang, Qianhui Wu, Chin-Yew Lin, Yuqing Yang, and Lili Qiu. Llmlingua: Compressing prompts for accelerated inference of large language models. *arXiv preprint arXiv:2310.05736*, 2023.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. Billion-scale similarity search with gpus. *IEEE Transactions on Big Data*, 7(3):535–547, 2019.

Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models. *arXiv preprint arXiv:2001.08361*, 2020.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*, 2020.

Omar Khattab and Matei Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over bert. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48, 2020.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. Latent retrieval for weakly supervised open domain question answering. *arXiv preprint arXiv:1906.00300*, 2019.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. Retrieval-augmented generation for knowledge-intensive nlp tasks. *Advances in Neural Information Processing Systems*, 33:9459–9474, 2020.

Junnan Li, Dongxu Li, Silvio Savarese, and Steven Hoi. Blip-2: Bootstrapping language-image pre-training with frozen image encoders and large language models. In *International conference on machine learning*, pages 19730–19742. PMLR, 2023.

Liunian Harold Li, Mark Yatskar, Da Yin, Cho-Jui Hsieh, and Kai-Wei Chang. Visualbert: A simple and performant baseline for vision and language. *arXiv preprint arXiv:1908.03557*, 2019.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

Weizhe Lin and Bill Byrne. Retrieval augmented visual question answering with outside knowledge. *arXiv preprint arXiv:2210.03809*, 2022.

Weizhe Lin, Jinghong Chen, Jingbiao Mei, Alexandru Coca, and Bill Byrne. Fine-grained late-interaction multi-modal retrieval for retrieval augmented visual question answering. *Advances in Neural Information Processing Systems*, 36, 2024a.

Weizhe Lin, Jingbiao Mei, Jinghong Chen, and Bill Byrne. Preflmr: Scaling up fine-grained late-interaction multi-modal retrievers. *arXiv preprint arXiv:2402.08327*, 2024b.

Yuanze Lin, Yujia Xie, Dongdong Chen, Yichong Xu, Chenguang Zhu, and Lu Yuan. Revive: Regional visual representation matters in knowledge-based visual question answering. *Advances in Neural Information Processing Systems*, 35:10560–10571, 2022.

Haotian Liu, Chunyuan Li, Qingyang Wu, and Yong Jae Lee. Visual instruction tuning. *Advances in neural information processing systems*, 36, 2024a.

Nelson F Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173, 2024b.

Jiasen Lu, Dhruv Batra, Devi Parikh, and Stefan Lee. Vilbert: Pretraining task-agnostic visiolinguistic representations for vision-and-language tasks. *Advances in neural information processing systems*, 32, 2019.

Yu A Malkov and Dmitry A Yashunin. Efficient and robust approximate nearest neighbor search using hierarchical navigable small world graphs. *IEEE transactions on pattern analysis and machine intelligence*, 42(4):824–836, 2018.

Kenneth Marino, Mohammad Rastegari, Ali Farhadi, and Roozbeh Mottaghi. Ok-vqa: A visual question answering benchmark requiring external knowledge. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 3195–3204, 2019.

Irina Matveeva, Chris Burges, Timo Burkard, Andy Laucius, and Leon Wong. High accuracy retrieval with multiple nested ranker. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 437–444, 2006.

Thomas Mensink, Jasper Uijlings, Lluis Castrejon, Arushi Goel, Felipe Cadar, Howard Zhou, Fei Sha, André Araujo, and Vittorio Ferrari. Encyclopedic vqa: Visual questions about detailed properties of fine-grained categories. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3113–3124, 2023.

Rodrigo Nogueira and Kyunghyun Cho. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*, 2019.

Rodrigo Nogueira, Wei Yang, Kyunghyun Cho, and Jimmy Lin. Multi-stage document ranking with bert. *arXiv preprint arXiv:1910.14424*, 2019.

Rodrigo Nogueira, Zhiying Jiang, and Jimmy Lin. Document ranking with a pretrained sequence-to-sequence model. *arXiv preprint arXiv:2003.06713*, 2020.

Ronak Pradeep, Rodrigo Nogueira, and Jimmy Lin. The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models. *arXiv preprint arXiv:2101.05667*, 2021.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, Gretchen Krueger, and Ilya Sutskever. Learning transferable visual models from natural language supervision. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 8748–8763. PMLR, 18–24 Jul 2021a. URL https://proceedings.mlr.press/v139/radford21a.html.

Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International conference on machine learning*, pages 8748–8763. PMLR, 2021b.

Ruiyang Ren, Yingqi Qu, Jing Liu, Wayne Xin Zhao, Qiaoqiao She, Hua Wu, Haifeng Wang, and Ji-Rong Wen. Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking. *arXiv preprint arXiv:2110.07367*, 2021.

Adam Roberts and Colin Raffel. Exploring transfer learning with t5: the text-to-text transfer transformer. *Google AI Blog*, 2020.

Stephen E Robertson and Nicholas J Belkin. Ranking in principle. *Journal of Documentation*, 34(2):93–100, 1978.

Stephen E Robertson, Steve Walker, Susan Jones, Micheline M Hancock-Beaulieu, Mike Gatford, et al. Okapi at trec-3. *Nist Special Publication Sp*, 109:109, 1995.

Leonard Salewski, A Sophia Koepke, Hendrik PA Lensch, and Zeynep Akata. Clevr-x: A visual reasoning dataset for natural language explanations. In *International Workshop on Extending Explainable AI Beyond Deep Models and Classifiers*, pages 69–88. Springer, 2020.

Keshav Santhanam, Omar Khattab, Jon Saad-Falcon, Christopher Potts, and Matei Zaharia. Colbertv2: Effective and efficient retrieval via lightweight late interaction. *arXiv preprint arXiv:2112.01488*, 2021.

Keshav Santhanam, Omar Khattab, Christopher Potts, and Matei Zaharia. Plaid: an efficient engine for late interaction retrieval. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 1747–1756, 2022.

Sanket Shah, Anand Mishra, Naganand Yadati, and Partha Pratim Talukdar. Kvqa: Knowledge-aware visual question answering. In *Proceedings of the AAAI conference on artificial intelligence*, volume 33, pages 8876–8884, 2019.

Piyush Sharma, Nan Ding, Sebastian Goodman, and Radu Soricut. Conceptual captions: A cleaned, hypernymed, image alt-text dataset for automatic image captioning. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2556–2565, 2018.

Anshumali Shrivastava and Ping Li. Asymmetric lsh (alsh) for sublinear time maximum inner product search (mips). *Advances in neural information processing systems*, 27, 2014.

Karen Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of documentation*, 28(1):11–21, 1972.

Krishna Srinivasan, Karthik Raman, Jiecao Chen, Michael Bendersky, and Marc Najork. Wit: Wikipedia-based image text dataset for multimodal multilingual machine learning. In *Proceedings of the 44th international ACM SIGIR conference on research and development in information retrieval*, pages 2443–2449, 2021.

Weijie Su, Xizhou Zhu, Yue Cao, Bin Li, Lewei Lu, Furu Wei, and Jifeng Dai. Vl-bert: Pre-training of generic visual-linguistic representations. *arXiv preprint arXiv:1908.08530*, 2019.

Hao Tan and Mohit Bansal. Lxmert: Learning cross-modality encoder representations from transformers. *arXiv preprint arXiv:1908.07490*, 2019.

Gemini Team, Rohan Anil, Sebastian Borgeaud, Yonghui Wu, Jean-Baptiste Alayrac, Jiahui Yu, Radu Soricut, Johan Schalkwyk, Andrew M Dai, Anja Hauth, et al. Gemini: a family of highly capable multimodal models. *arXiv preprint arXiv:2312.11805*, 2023.

Maria Tsimpoukelli, Jacob L Menick, Serkan Cabi, SM Eslami, Oriol Vinyals, and Felix Hill. Multimodal few-shot learning with frozen language models. *Advances in Neural Information Processing Systems*, 34:200–212, 2021.

Haoyang Wen, Honglei Zhuang, Hamed Zamani, Alexander Hauptmann, and Michael Bendersky. Multimodal reranking for knowledge-intensive visual question answering. *arXiv preprint arXiv:2407.12277*, 2024.

Fangyuan Xu, Weijia Shi, and Eunsol Choi. Recomp: Improving retrieval-augmented lms with compression and selective augmentation. *arXiv preprint arXiv:2310.04408*, 2023.

Zhengyuan Yang, Zhe Gan, Jianfeng Wang, Xiaowei Hu, Yumao Lu, Zicheng Liu, and Lijuan Wang. An empirical study of gpt-3 for few-shot knowledge-based vqa. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 36, pages 3081–3089, 2022.

Andrew Yates, Rodrigo Nogueira, and Jimmy Lin. Pretrained transformers for text ranking: Bert and beyond. In *Proceedings of the 14th ACM International Conference on web search and data mining*, pages 1154–1156, 2021.

Li-Ming Zhan, Bo Liu, Lu Fan, Jiaxin Chen, and Xiao-Ming Wu. Medical visual question answering via conditional reasoning. In *Proceedings of the 28th ACM International Conference on Multimedia*, pages 2345–2354, 2020.

Susan Zhang, Stephen Roller, Naman Goyal, Mikel Artetxe, Moya Chen, Shuohui Chen, Christopher Dewan, Mona Diab, Xian Li, Xi Victoria Lin, et al. Opt: Open pre-trained transformer language models. *arXiv preprint arXiv:2205.01068*, 2022.

Tianjun Zhang, Shishir G Patil, Naman Jain, Sheng Shen, Matei Zaharia, Ion Stoica, and Joseph E Gonzalez. Raft: Adapting language model to domain specific rag. *arXiv preprint arXiv:2403.10131*, 2024.

Honglei Zhuang, Zhen Qin, Rolf Jagerman, Kai Hui, Ji Ma, Jing Lu, Jianmo Ni, Xuanhui Wang, and Michael Bendersky. Rankt5: Fine-tuning t5 for text ranking with ranking losses. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2308–2313, 2023.

L. Zou, S. Zhang, H. Cai, D. Ma, S. Cheng, S. Wang, D. Shi, Z. Cheng, and D. Yin. Pre-trained language model based ranking in baidu search. In *Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining*, pages 4014–4022, August 2021.

Appendix A

Tabular Results

A.1 Pointwise Results

Category	Reranker	Recall@5	Recall@10	Recall@20	Recall@50	Time per Query (s)	Parameters
Raw Retrieval	*Raw Retrieval*	*0.2346*	*0.3231*	*0.4125*	*0.5394*	-	-
monoPreFLMR	**monoPreFLMR-B**	**0.2836**	**0.3778**	**0.4644**	**0.5749**	**1.40**	**248M**
	monoPreFLMR-L	0.2711	0.3576	0.4653	0.5701	2.58	465M
monoBLIP-2	monoBLIP-2 Flan-T5	0.3605	0.4528	0.5442	0.6173	9.52	3.9B
	monoBLIP-2 Opt	**0.3980**	**0.4798**	**0.5528**	**0.6192**	**12.64**	**3.7B**
ModPreFLMR	ModPreFLMR BERT	0.2519	0.3462	0.4375	0.5740	0.24	46M
	ModPreFLMR IB	**0.2519**	**0.3480**	**0.4480**	**0.5548**	**0.13**	**47M**

Table A.1 **Appendix:** OKVQA Reranker Recall

Category	Reranker	Recall@5	Recall@10	Recall@20	Recall@50	Time per Query (s)	Parameters
Raw Retrieval	*Raw Retrieval*	*0.2048*	*0.3105*	*0.4355*	*0.5932*	-	-
monoPreFLMR	**monoPreFLMR-B** *(15000 steps)*	**0.0990**	**0.1548**	**0.2615**	**0.5442**	**1.40**	**248M**
	monoPreFLMR-B *(6000 steps)*	0.0750	0.1317	0.2307	0.4749	2.58	465M
	monoPreFLMR-L *(6000 steps)*	0.0548	0.1048	0.2076	0.4682	2.58	465M
monoBLIP-2	**monoBLIP-2 Flan-T5**	**0.3086**	**0.4134**	**0.5124**	**0.6663**	**9.52**	**3.9B**
	monoBLIP-2 Opt	0.2663	0.3567	0.4798	0.6634	12.64	3.7B
ModPreFLMR	**ModPreFLMR BERT**	**0.1759**	**0.2788**	**0.4019**	**0.5990**	**0.24**	**46M**
	ModPreFLMR IB	0.1403	0.2365	0.3413	0.5759	0.13	47M

Table A.2 **Appendix:** EVQA Reranker Recall

A.2 Listwise Results

Dataset	Reranker	Loss	Recall@5	Recall@10	Recall@20	Recall@50
OKVQA	**monoPreFLMR-B**	***Pointwise***	***0.2836***	***0.3778***	***0.4644***	***0.5749***
		Listwise	0.2826	0.3778	0.4663	0.5711
	monoBLIP-2 Opt	*Text Gen Pointwise*	*0.3980*	*0.4798*	*0.5528*	*0.6192*
		Pointwise	**0.4307**	**0.5038**	**0.5730**	**0.6182**
		Listwise	0.3942	0.4807	0.5548	0.6153
	ModPreFLMR IB	***Pointwise***	***0.2519***	***0.3480***	***0.4480***	***0.5548***
		Listwise	0.2346	0.3115	0.4182	0.5480
EVQA	**monoBLIP-2 Flan-T5**	*Text Gen Pointwise*	*0.3086*	*0.4134*	*0.5124*	*0.6663*
		Pointwise	**0.3394**	**0.4519**	**0.5721**	**0.6961**
		Listwise	0.3336	0.4365	0.5528	0.6942
	ModPreFLMR BERT	***Pointwise***	***0.1759***	***0.2788***	***0.4019***	***0.5990***
		Listwise	0.1721	0.2653	0.3884	0.5961

Table A.3 **Appendix:** Reranker Recall by Loss Function

A.3 Fine-Tuning on Retrieved Documents Results

		Full Corpus			Raw Retrieval		
Dataset	**Reranker**	**Train Loss**	**Val Loss**	**Recall@5**	**Train Loss**	**Val Loss**	**Recall@5**
OKVQA	**monoPreFLMR-B**	0.0226	3.062	0.2836	**0.1266**	**0.1018**	**0.3259**
	monoBLIP-2 OPT	**0.1133**	**1.780**	**0.4307**	0.1009	0.08	0.4134
	ModPreFLMR IB	0.0442	3.458	0.2519	**0.1072**	**0.1148**	**0.2769**
EVQA	**monoBLIP-2 Flan-T5**	0.0304	2.244	0.3394	**0.0278**	**0.0476**	**0.3634**
	ModPreFLMR BERT	**0.0171**	**4.686**	**0.1759**	0.0348	0.0479	0.1192

Table A.4 **Appendix:** Fine-Tuning on Full Corpus vs Raw Retrieval

Dataset	**Reranker**	**Training Samples**	**Recall@5**	**Recall@10**	**Recall@20**	**Recall@50**
OKVQA	**monoPreFLMR-B**	*Full Corpus*	*0.2836*	*0.3778*	*0.4644*	*0.5749*
		Raw Retrieval	0.3259	0.4009	0.4903	0.5826
		Upsampled Retrieval	0.1394	0.2125	0.3355	0.5163
		Listwise Retrieval	**0.3432**	**0.4326**	**0.5000**	**0.5875**
	monoBLIP-2 OPT	*Full Corpus*	*0.4307*	*0.5038*	*0.5730*	*0.6182*
		Raw Retrieval	0.4134	0.4932	0.5625	0.6163
		Upsampled Retrieval	0.1375	0.4932	0.5625	0.6163
		Listwise Retrieval	**0.4461**	**0.5067**	**0.5653**	**0.6201**
	ModPreFLMR IB	*Full Corpus*	*0.2519*	*0.3480*	*0.4480*	*0.5548*
		Raw Retrieval	0.2769	0.3538	0.4403	0.5634
		Upsampled Retrieval	0.1125	0.1846	0.2923	0.5634
		Listwise Retrieval	**0.2913**	**0.3855**	**0.4615**	**0.5711**
EVQA	**monoBLIP-2 Flan-T5**	*Full Corpus*	*0.3394*	*0.4519*	*0.5721*	*0.6961*
		Raw Retrieval	0.3634	0.4759	0.5903	0.6903
		Upsampled Retrieval	**0.3826**	**0.4932**	**0.5865**	**0.7057**
		Listwise Retrieval	0.3653	0.4625	0.5778	0.6942
	ModPreFLMR BERT	*Full Corpus*	*0.1759*	*0.2788*	*0.4019*	*0.5990*
		Raw Retrieval	0.1192	0.2221	0.3653	0.5682
		Upsampled Retrieval	**0.2105**	**0.2961**	**0.4336**	**0.6201**
		Listwise Retrieval	0.1990	0.2971	0.4086	0.5769

Table A.5 **Appendix:** Reranker Recall by Training Samples

A.4 Ablation Results

Dataset	Model	Activation	Recall@5	Recall@10	Recall@20	Recall@50
OKVQA	**monoPreFLMR-B**	***Sigmoid***	***0.2635***	***0.3500***	***0.4354***	***0.5625***
		Softmax	0.2427	0.3240	0.4271	0.5583

Table A.6 **Appendix:** Reranker Recall by Activation Function

Dataset	Model	Recall@5	Recall@10	Recall@20	Recall@50
OKVQA	***monoPreFLMR-B***	***0.2836***	***0.3778***	***0.4644***	***0.5749***
	monoPreFLMR-L	0.2711	0.3576	0.4653	0.5701
	monoBERT	0.2798	0.3557	0.4384	0.5692
EVQA	***monoPreFLMR-B***	***0.0750***	***0.1317***	***0.2307***	**0.4749**
	monoPreFLMR-L	0.0548	0.1048	0.2076	0.4682
	monoBERT	0.0480	0.1038	0.2096	0.4807

Table A.7 **Appendix:** Reranker Recall by Vision Model

Data	Model	Cross-Encoder Layers	Recall@5	Recall@10	Recall@20	Recall@50
OKVQA	**monoPreFLMR-B**	***1 Layer***	***0.2836***	***0.3778***	***0.4644***	***0.5749***
		3 Layer	0.2548	0.3451	0.4596	0.5903
	ModPreFLMR IB	***5 Layer***	***0.2519***	***0.3480***	***0.4480***	***0.5548***
		8 Layer	0.2173	0.3173	0.4096	0.5432

Table A.8 **Appendix:** Reranker Recall by Cross-Encoder Layers

Data	Model	Vision Model	Recall@5	Recall@10	Recall@20	Recall@50
OKVQA	**monoPreFLMR-B**	***Frozen***	***0.2836***	***0.3778***	***0.4644***	***0.5749***
		Trainable	0.2653	0.3701	0.4567	0.5701

Table A.9 **Appendix:** Reranker Recall by Freezing Vision Model

Dataset	Reranker	Retrieval List Size (D)					
		5	**10**	**25**	**50**	**75**	**100**
OKVQA	**monoPreFLMR-B**	0.2346	0.2798	0.3433	0.3433	**0.3442**	0.3433
	monoBLIP-2 Opt	0.2346	0.2981	0.3712	0.4000	0.4327	**0.4462**
	ModPreFLMR IB	0.2346	0.2798	0.3029	**0.3077**	0.2962	0.2913
EVQA	**monoPreFLMR-B**	**0.2048**	0.1721	0.1317	0.1106	0.1019	0.0990
	monoBLIP-2 Flan-T5	0.2048	0.2875	0.3471	0.3663	0.3817	**0.3827**
	ModPreFLMR BERT	0.2048	**0.2240**	0.2163	0.2173	0.2115	0.2106

Table A.10 **Appendix:** Recall@5 vs Retrieval List Size