Comparative Analysis of Lion and AdamW Optimizers for Cross-Encoder Reranking with MiniLM, GTE, and ModernBERT

A thesis submitted in partial fulfillment of the requirements for the award of the degree of

MASTER OF TECHNOLOGY

By:-	*Enrollment No.*
SHAHIL KUMAR	MML2023008

Under the Supervision of

Dr. Muneendra Ojha

to the

Department of Information Technology

भारतीय सूचना प्रौद्योगिकी संस्थान, इलाहाबाद

Indian Institute of Information Technology, Allahabad

May , 2025

भारतीय सूचना प्रौद्योगिकी संस्थान इलाहाबाद
Indian Institute of Information Technology Allahabad
An Institute of National Importance by Act of Parliament
Deoghat Jhalwa, Prayagraj 211015 (U.P.) India
Ph: 0532-2922025, 2922067; Web: www.iiita.ac.in; Email: contact@iiita.ac.in

CANDIDATE DECLARATION

I hereby declare that work presented in the report entitled **Comparative Analysis of Lion and AdamW Optimizers for Cross-Encoder Reranking with MiniLM, GTE, and ModernBERT**, submitted towards the fulfillment of MASTERS THESIS report of M.Tech at Indian Institute of Information Technology Allahabad, is an authenticated original work carried out under supervision of **Dr. Muneendra Ojha**. Due Acknowledgements have been made in the text to all other material used. the project was done in full compliance with the requirements and constraints of the prescribed curriculum.

Shahil Kumar - MML2023008

भारतीय सूचना प्रौद्योगिकी संस्थान इलाहाबाद

Indian Institute of Information Technology Allahabad

An Institute of National Importance by Act of Parliament

Deoghat Jhalwa, Prayagraj 211015 (U.P.) India

Ph: 0532-2922025, 2922067; Web: www.iiita.ac.in; Email: contact@iiita.ac.in

CERTIFICATE FROM SUPERVISORS

It is certified that the work contained in the thesis titled "**Comparative Analysis of Lion and AdamW Optimizers for Cross-Encoder Reranking with MiniLM, GTE, and ModernBERT**" by **Shahil kumar** has been carried out under supervision of **Dr. Muneendra Ojha** and that this work has not been submitted elsewhere for a degree.

Dr. Muneendra Ojha
Department of Information Technology
IIIT Allahabad

भारतीय सूचना प्रौद्योगिकी संस्थान इलाहाबाद
Indian Institute of Information Technology Allahabad
An Institute of National Importance by Act of Parliament
Deoghat Jhalwa, Prayagraj 211015 (U.P.) India
Ph: 0532-2922025, 2922067; Web: www.iiita.ac.in; Email: contact@iiita.ac.in

CERTIFICATE OF APPROVAL

This thesis entitled **Comparative Analysis of Lion and AdamW Optimizers for Cross-Encoder Reranking with MiniLM, GTE, and ModernBERT** by **Shahil Kumar** (MML2023008) is approved for the degree of Master's thesis at IIIT Allahabad It is understood that by this approval, the undersigned does not necessarily endorse or approve any statement made, opinion expressed, or conclusion drawn therein but approves the thesis only for the purpose for which it is submitted."

Signature and name of the committee members (on final examination and approval of the thesis):

1. Dr. Muneendra Ojha

2. Dr. Kavindra Kandpal

3. Dr. Anand Kumar Tiwari

Dean(A&R)

reventh

ORIGINALITY REPORT

6%	5%	5%	2%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.researchgate.net Internet Source	1%
2	ethesis.nitrkl.ac.in Internet Source	<1%
3	cdn.slideserve.com Internet Source	<1%
4	www.uspaquatic.library.usp.ac.fj Internet Source	<1%
5	umpir.ump.edu.my Internet Source	<1%
6	Jialu Yin, Jia Yuan, Heng You, Peng Wang, Shushan Qiao. "A fast and low-power level shifter for multi-supply voltage designs", IEICE Electronics Express, 2020 Publication	<1%
7	ijiemr.org Internet Source	<1%
8	Submitted to Visvesvaraya Technological University, Belagavi	<1%

Student Paper

9	Abhay S. Vidhyadharan, Aiswarya Satheesh, Kilari Pragnaa, Sanjay Vidhyadharan. "High-Speed and Area-Efficient CMOS and CNFET-Based Level-Shifters", Circuits, Systems, and Signal Processing, 2022 Publication	<1 %
10	PramodKumar Aylapogu, R.V. Prasad Bhookya, D. Venkatachari, Kalivaraprasad. B. "Dual Current Mirror Technique Based Energy Efficient 50mV to 1V Voltage Level Shifter", 2022 International Conference on Electronics and Renewable Systems (ICEARS), 2022 Publication	<1 %
11	Submitted to University of Technology, Sydney Student Paper	<1 %
12	pt.scribd.com Internet Source	<1 %
13	V J Arulkarthick, Rajendran Selvakumar, Chakrapani Arvind, Kannan Srihari. "High performance contention-eased full-swing level converter for multi-supply voltage systems", Sādhanā, 2021 Publication	<1 %
14	Suprbha Kumari, Rita Mahajan, Deepak Bagai. "Comparative Analysis of Power and Delay	<1 %

भारतीय सूचना प्रौद्योगिकी संस्थान इलाहाबाद

Indian Institute of Information Technology Allahabad

An Institute of National Importance by Act of Parliament

Deoghat Jhalwa, Prayagraj 211015 (U.P.) India

Ph: 0532-2922025, 2922067; Web: www.iiita.ac.in; Email: contact@iiita.ac.in

ACKNOWLEDGEMENT

I am thankful to my project supervisor, **Dr. Muneendra Ojha** for the guidance, support, and invaluable feedback throughout the research process. Their expertise, encouragement, and patience have been instrumental in the completion of this thesis.

I am grateful to Dr. K.P. Singh, Head of the Department of Information Technology, and also my panel members Dr Kavindra Kandpal and Prof. Anand Kumar for their guidance and support.

I would also like to thank the Indian Institute of Information Technology, Allahabad for providing the necessary resources and facilities to conduct this research. I am deeply grateful to the participants who generously shared their time and personal information to make this study possible.

Shahil Kumar - MML2023008

ABSTRACT

Modern information retrieval systems often employ a two-stage pipeline consisting of an efficient initial retrieval stage followed by a more computationally intensive reranking stage. Cross-encoder models have demonstrated state-of-the-art effectiveness for the reranking task due to their ability to perform deep, contextualized analysis of query-document pairs. The choice of optimizer during the fine-tuning phase can significantly impact the final performance and training efficiency of these models. This paper investigates the impact of using the recently proposed Lion optimizer compared to the widely used AdamW optimizer for fine-tuning cross-encoder rerankers. We fine-tune three distinct transformer models, 'microsoft/MiniLM-L12-H384-uncased', 'Alibaba-NLP/gte-multilingual-base', and 'answerdotai/ModernBERT-base', on the MS MARCO passage ranking dataset using both optimizers. Notably, GTE and ModernBERT support longer context lengths (8192 tokens). The effectiveness of the resulting models is evaluated on the TREC 2019 Deep Learning Track passage ranking task and the MS MARCO development set (for MRR@10). Our experiments, facilitated by the Modal cloud computing platform for GPU resource management, show comparative results across three training epochs. ModernBERT trained with Lion achieved the highest NDCG@10 (0.7225) and MAP (0.5121) on TREC DL 2019, while MiniLM trained with Lion tied with ModernBERT with Lion on MRR@10 (0.5988) on MS MARCO dev. We analyze the performance trends based on standard IR metrics, providing insights into the relative effectiveness of Lion versus AdamW for different model architectures and training configurations in the context of passage reranking.

Table Of Contents

LIST OF FIGURES

LIST OF TABLES

Chapter 1

Introduction

Information Retrieval (IR) systems enable users to find relevant information from vast document collections. Modern IR systems employ a two-stage pipeline: efficient initial retrieval (BM25 [30] or dense vector retrieval [13]) followed by sophisticated reranking for precision optimization.

Cross-encoder models represent the state-of-the-art for reranking [24, 26] due to their ability to model deep query-document interactions. Unlike bi-encoders that process queries and documents independently, cross-encoders simultaneously analyze query-document pairs, enabling rich token-level interactions for accurate relevance estimation.

The effectiveness of cross-encoder models depends heavily on optimization strategies during fine-tuning. While AdamW [20] has been the standard choice for transformer training, recent developments in optimization algorithms, particularly the Lion optimizer [3], have shown promising results across various domains, motivating investigation into their applicability for information retrieval tasks.

1.1 Cross-Encoder Models for Information Retrieval

Cross-encoders process query-document pairs by concatenating them with special tokens ([CLS] query [SEP] document [SEP]) and feeding the combined sequence through a transformer architecture [7]. The [CLS] token representation predicts relevance scores through a linear layer with sigmoid activation.

Modern implementations leverage various transformer architectures with distinct characteristics:

- **MiniLM-L12-H384**: Distilled BERT variant optimized for efficiency [35] (512 tokens context)
- **GTE-multilingual-base**: Extended context model (8192 tokens) with multilingual training [16]
- **ModernBERT-base**: State-of-the-art architecture with RoPE [32], Flash Attention [6], and GeGLU [31] [36]

1.2 Optimization Strategies

Cross-encoder effectiveness depends on optimization strategies during fine-tuning. While AdamW [20] has been the standard choice, the Lion optimizer [3] presents a novel approach using sign-based momentum: `update = sign(momentum) * learning_rate`. This simplification offers potential memory efficiency and different convergence characteristics compared to traditional adaptive methods.

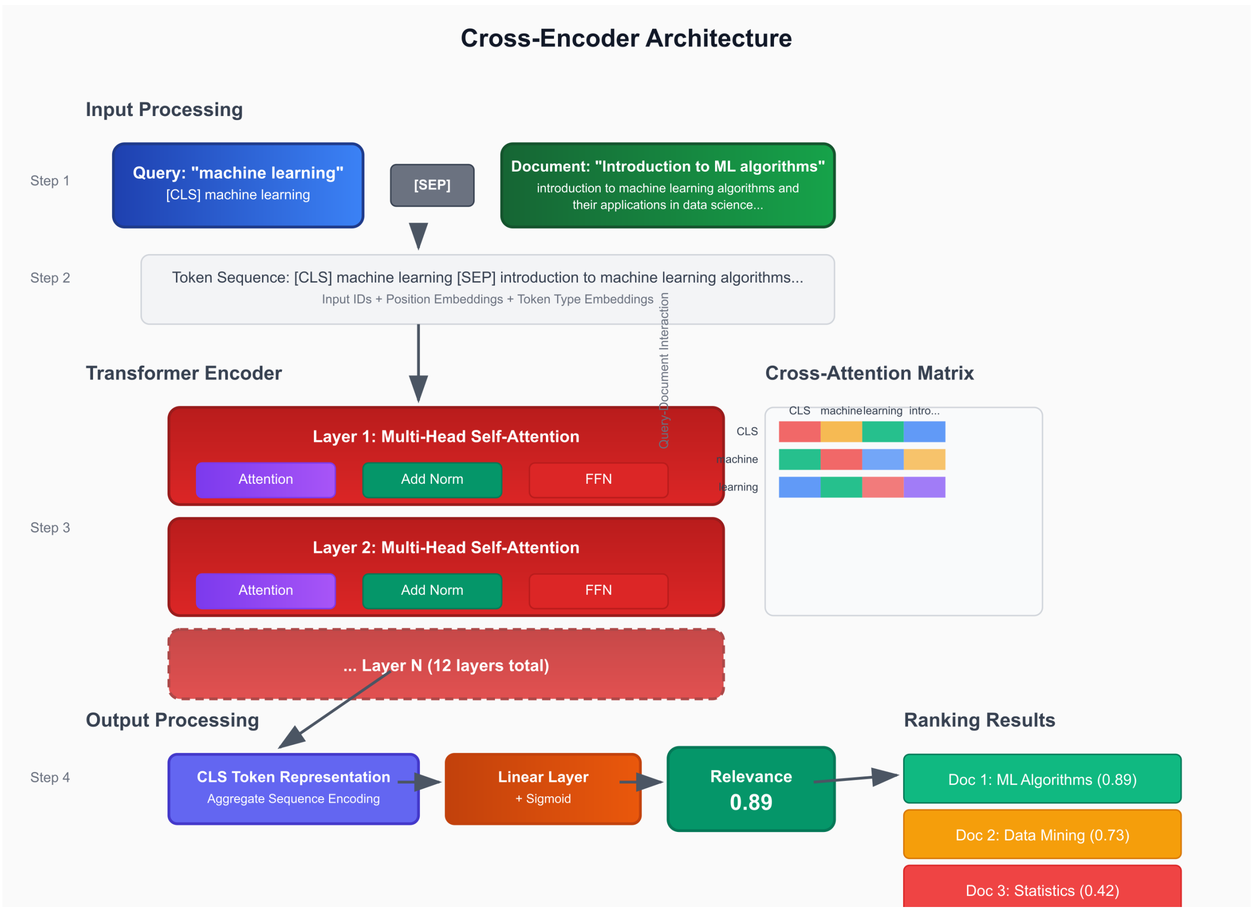
Cross-Encoder Architecture
Input Processing
Step 1
Query: "machine learning"
[CLS] machine learning
[SEP]
Document: "Introduction to ML algorithms"
introduction to machine learning algorithms and their applications in data science...
Step 2
Token Sequence: [CLS] machine learning [SEP] introduction to machine learning algorithms...
Input IDs + Position Embeddings + Token Type Embeddings
Query-Document Interaction
Transformer Encoder
Cross-Attention Matrix
CLS
machine
learning
intro...
Step 3
Layer 1: Multi-Head Self-Attention
Attention
Add Norm
FFN
Layer 2: Multi-Head Self-Attention
Attention
Add Norm
FFN
... Layer N (12 layers total)
Output Processing
Ranking Results
Step 4
CLS Token Representation
Aggregate Sequence Encoding
Linear Layer
+ Sigmoid
Relevance
0.89
Doc 1: ML Algorithms (0.89)
Doc 2: Data Mining (0.73)
Doc 3: Statistics (0.42)

FIGURE 1.1: Cross-Encoder Architecture for Query-Document Reranking.

1.3 Research Objectives

This research conducts a comprehensive comparative analysis of Lion and AdamW optimizers for cross-encoder models in information retrieval. Our primary objectives are:

1. **Systematic Evaluation**: Compare Lion and AdamW across three transformer architectures (MiniLM, GTE, ModernBERT)
2. **Comprehensive Benchmarking**: Evaluate performance using standard IR metrics on TREC DL 2019 and MS MARCO
3. **Training Dynamics Analysis**: Investigate convergence patterns and stability across training epochs
4. **Practical Guidelines**: Provide evidence-based recommendations for optimizer selection

1.3.1 Expected Contributions

- Empirical evidence comparing Lion and AdamW optimizers across multiple architectures
- Optimization guidelines for cross-encoder training in information retrieval
- Analysis of training efficiency and convergence characteristics

1.4 Methodology Overview

We employ systematic experimentation across three transformer architectures representing different performance-efficiency trade-offs: MiniLM (efficient baseline), GTE (extended context), and ModernBERT (state-of-the-art). Experiments utilize standardized datasets (MS MARCO [23] for training, TREC DL 2019 [4] for evaluation) with cloud infrastructure (Modal platform [21], NVIDIA A100-80GB GPUs) ensuring reproducible results. Comprehensive tracking through Weights & Biases [2] enables detailed training dynamics analysis.

1.5 Thesis Organization

This thesis systematically presents research methodology, experimental results, and implications: **Chapter 2**: Literature review on cross-encoder architectures and optimization algorithms **Chapter 3**: Research gaps and proposed comparative evaluation approach **Chapter 4**: Experimental methodology and implementation details **Chapter 5**: Comprehensive results with training dynamics and statistical analysis **Chapter 6**: Key findings, implications, and future research directions

The thesis contributes empirical insights into optimizer selection for cross-encoder architectures with implications for search engines, question-answering systems, and information retrieval applications where ranking quality is paramount.

Chapter 2

Literature Review

This chapter reviews key literature on cross-encoder models, optimization algorithms, and evaluation methodologies for information retrieval.

2.1 Neural Information Retrieval Evolution

Information retrieval has transformed from traditional term-based matching (TF-IDF, BM25 [30]) to neural approaches that capture semantic relationships. Dense retrieval methods like DPR [13] and ANCE [38] encode queries and documents independently, while cross-encoders enable richer query-document interactions.

2.2 Cross-Encoder Architectures

Cross-encoders leverage transformer architectures for document reranking. Nogueira and Cho [24] demonstrated BERT's effectiveness for reranking by concatenating queries and documents with special tokens. The [CLS] representation predicts relevance scores through fine-tuning on labeled data.

Modern architectures include:

- **MiniLM [35]**: Distilled BERT variant for efficiency
- **GTE [16]**: Multilingual model with extended context (8192 tokens)
- **ModernBERT [36]**: Advanced architecture with RoPE, Flash Attention, GeGLU

2.3 Optimization Algorithms

2.3.1 Traditional Optimizers

Adam [15] combines momentum with adaptive learning rates using first and second moment estimates. AdamW [20] improved generalization by decoupling weight decay from gradient updates, becoming the standard for transformer training.

2.3.2 Lion Optimizer

Lion [3] represents a paradigm shift using sign-based momentum: `update = sign(momentum) * learning_rate`. This simplified approach reduces memory requirements while maintaining competitive performance across vision tasks.

2.4 Evaluation Benchmarks

Standard IR evaluation relies on:

- **MS MARCO [23]**: Large-scale passage ranking dataset
- **TREC DL 2019 [4]**: Deep learning track for passage ranking
- **Metrics**: nDCG@10, MRR@10, MAP measure ranking quality

2.5 Research Gaps

While cross-encoder effectiveness is well-established, systematic investigation of optimization strategies remains limited. Most studies assume AdamW without exploring alternatives like Lion, particularly for information retrieval tasks where ranking precision is critical.

2.5.1 Modern Transformer Architectures

Recent developments in transformer architectures have introduced models specifically designed for improved efficiency and longer context processing. The General Text Embeddings (GTE) family [16] demonstrated that models trained on diverse, multilingual corpora could achieve strong performance across various text understanding tasks, including retrieval and reranking.

ModernBERT [36] represents the current state-of-the-art in encoder-only transformers, incorporating several architectural innovations:

- **Rotary Positional Embeddings (RoPE)** [32]: Enable effective handling of longer sequences by providing more flexible positional encoding.
- **Flash Attention** [6]: Improves memory efficiency and computational speed for attention operations.
- **GeGLU Activation Functions** [31]: Enhance model expressiveness while maintaining computational efficiency.

These architectural improvements enable ModernBERT to process sequences up to 8192 tokens efficiently, making it particularly suitable for long document processing tasks.

2.6 Optimization Algorithms for Deep Learning

The success of neural information retrieval models depends heavily on the optimization algorithms used during training. The choice of optimizer affects convergence speed, final performance, and training stability. This section reviews the evolution of optimization algorithms and their specific applications to transformer-based models.

2.6.1 Classical Optimization Methods

Stochastic Gradient Descent (SGD) remains a fundamental optimization algorithm, providing theoretical guarantees and interpretable behavior. However, the challenges of training deep neural networks - including vanishing gradients, saddle points, and varying curvature across parameter space - motivated the development of adaptive optimization methods.

The introduction of momentum to SGD addressed some limitations by accumulating gradients across iterations, helping optimization navigate through areas of high curvature and accelerate convergence in consistent directions. However, SGD with momentum still struggled with the varying scales of different parameters and the need for careful learning rate tuning.

2.6.2 Adaptive Learning Rate Methods

AdaGrad [8] introduced the concept of adaptive learning rates, automatically adjusting step sizes based on historical gradient information. By maintaining per-parameter learning rates inversely proportional to the square root of accumulated squared gradients, AdaGrad could handle sparse features effectively and reduce the need for manual learning rate tuning.

RMSprop [34] addressed AdaGrad's aggressive learning rate decay by using exponential moving averages instead of accumulating all historical gradients. This modification prevented the learning rate from becoming too small too quickly, enabling continued learning throughout training.

Adam [15] combined the benefits of momentum-based methods with adaptive learning rates, maintaining separate exponentially decaying averages for both gradients (first moment) and squared gradients (second moment). The Adam optimizer became widely adopted due to its robustness across different architectures and tasks, requiring minimal hyperparameter tuning in many scenarios.

2.6.3 AdamW and Weight Decay Regularization

AdamW [20] represented a significant improvement over Adam by decoupling weight decay regularization from the adaptive learning rate mechanism. The key insight was that traditional L2 regularization in Adam led to suboptimal behavior because the adaptive learning rate scaling affected both gradient and regularization terms.

By separating weight decay from gradient-based updates, AdamW achieved better generalization performance and more stable training dynamics. The decoupled formulation:

$$\theta_{t+1} = \theta_t - \eta \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda \theta_t \right) \tag{2.1}$$

where λ represents the weight decay coefficient applied directly to parameters, independent of the adaptive learning rate scaling.

AdamW quickly became the standard optimizer for transformer-based models, demonstrating superior performance across various natural language processing tasks and establishing itself as the default choice for most deep learning applications.

2.6.4 Lion Optimizer: A Novel Approach

The Lion (EvoLved Sign mOmeNtum) optimizer [3] represents a departure from traditional adaptive optimization approaches. Developed through symbolic mathematics and program search, Lion utilizes a simplified update rule that relies primarily on the sign of momentum-based gradients.

The Lion update mechanism follows:

$$c_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{2.2}$$

$$\theta_t = \theta_{t-1} - \eta \left(\text{sign}(c_t) + \lambda \theta_{t-1}\right) \tag{2.3}$$

$$m_t = \beta_2 m_{t-1} + (1 - \beta_2) g_t \tag{2.4}$$

Chapter 3

Research Gap & Methodology

3.1 Research Gap

While AdamW has become standard for transformer training, the Lion optimizer's effectiveness for cross-encoder IR tasks remains unexplored. Key gaps include:

- **Novel Optimizer Evaluation**: Lion optimizer's memory efficiency and simplified updates haven't been systematically evaluated for cross-encoder training in IR contexts.
- **Architecture-Optimizer Interaction**: Different transformer architectures (MiniLM, GTE, ModernBERT) may respond differently to optimization strategies.
- **Training Efficiency Trade-offs**: Lion's claimed efficiency advantages need validation for computationally intensive cross-encoder training.
- **Comprehensive IR Benchmarking**: Systematic evaluation across standard IR datasets (MS MARCO, TREC DL) with multiple metrics is needed.

3.2 Methodology

This research systematically compares Lion and AdamW optimizers across three cross-encoder architectures for information retrieval tasks.

3.2.1 Model Selection

Three transformer architectures representing different efficiency-effectiveness points:

- **MiniLM-L12-H384**: Compact model (384 hidden dims, 12 layers) for efficiency
- **GTE-multilingual-base**: Multilingual model with 8192 token context support
- **ModernBERT-base**: State-of-the-art encoder with RoPE, Flash Attention, GeGLU

3.2.2 Experimental Design

Training Setup: Modal cloud platform with NVIDIA A100-80GB GPUs, batch size 16, 3 epochs, MS MARCO passage dataset.

Optimizer Configurations:

- **AdamW**: Learning rates 2e-5 (MiniLM, GTE), 2e-6 (ModernBERT), weight decay 0.01
- **Lion**: Same learning rates, =0.9, =0.99, sign-based momentum updates

Evaluation: TREC 2019 DL Track and MS MARCO dev set using NDCG@10, MAP, MRR@10 metrics.

Chapter 4

Cross-Encoder Architecture and Training Framework

4.1 Cross-Encoder Architecture for Information Retrieval

4.1.1 Architectural Overview

Cross-encoder models represent a significant advancement in neural information retrieval, enabling deep interaction modeling between queries and documents. Unlike bi-encoder approaches that process queries and documents independently, cross-encoders perform joint encoding, allowing for rich token-level interactions that lead to superior relevance estimation.

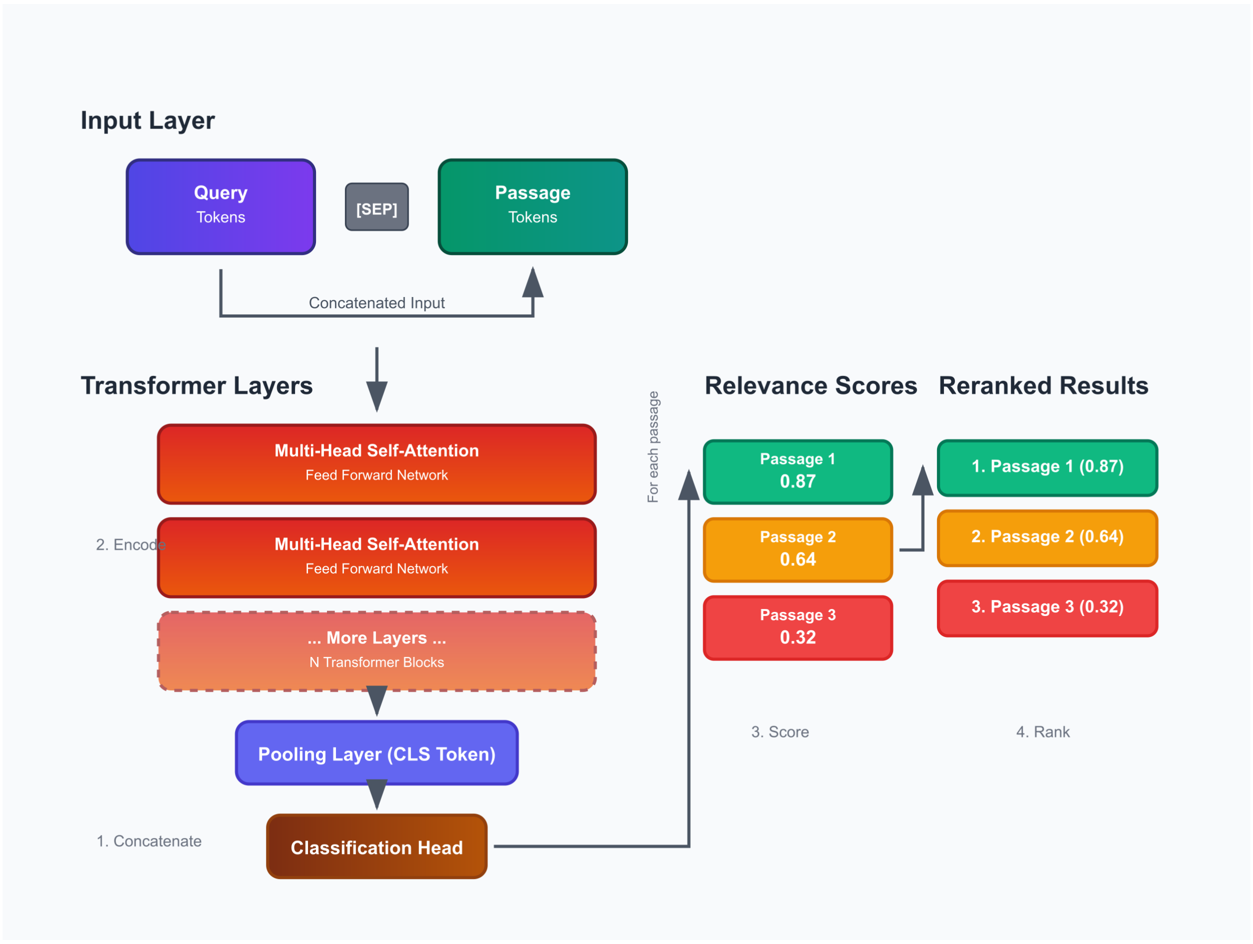

FIGURE 4.1: Cross-Encoder Architecture for Passage Reranking. The model processes concatenated query-passage pairs through transformer layers, producing relevance scores for reranking candidate documents.

The cross-encoder architecture follows a structured approach to relevance modeling:

1. **Input Formatting**: Query and passage text are concatenated with special tokens: [CLS] query [SEP] passage [SEP]
2. **Tokenization**: The input sequence is tokenized using subword tokenization (typically BERT-style WordPiece)
3. **Transformer Encoding**: The concatenated sequence passes through multiple transformer layers with self-attention mechanisms

4. **Relevance Prediction**: The [CLS] token representation is fed to a classification head producing a relevance score
5. **Score Normalization**: Sigmoid activation ensures scores fall within [0,1] range for ranking

4.1.2 Model Architecture Specifications

Three distinct transformer architectures were selected to represent different efficiency-effectiveness trade-offs:

MiniLM-L12-H384-uncased

MiniLM [35] represents an efficient architecture designed through knowledge distillation:

Architectural Parameters:

- Hidden dimensions: 384
- Number of layers: 12
- Attention heads: 12
- Maximum sequence length: 512 tokens
- Parameter count: 33M parameters
- Vocabulary size: 30,522 tokens

Design Philosophy: MiniLM achieves efficiency through knowledge distillation from larger teacher models while maintaining competitive performance. The reduced hidden dimensions and moderate layer count make it suitable for resource-constrained scenarios while preserving essential language understanding capabilities.

GTE-multilingual-base

The General Text Embeddings (GTE) model [16] extends capabilities to multilingual contexts with enhanced capacity:

Architectural Parameters:

- Hidden dimensions: 768
- Number of layers: 12
- Attention heads: 12
- Maximum sequence length: 8192 tokens
- Parameter count: 305M parameters
- Vocabulary size: 250,048 tokens
- Multilingual vocabulary support

Design Philosophy: GTE models are trained on diverse, multilingual corpora with multi-stage contrastive learning. The extended context length (8192 tokens) enables processing of longer documents, making it particularly suitable for comprehensive passage analysis and cross-lingual retrieval scenarios.

ModernBERT-base

ModernBERT [36] incorporates state-of-the-art architectural innovations for improved efficiency and effectiveness:

Architectural Parameters:

- Hidden dimensions: 768

- Number of layers: 22
- Attention heads: 12
- Maximum sequence length: 8192 tokens
- Parameter count: 149M parameters
- Vocabulary size: 50,368 tokens
- Advanced positional encoding and attention mechanisms

Architectural Innovations:

1. **Rotary Position Embedding (RoPE)** [32]: Enables effective handling of longer sequences through relative position encoding
2. **Flash Attention** [6]: Improves memory efficiency and computational speed for attention operations
3. **GeGLU Activation Functions** [31]: Enhances model expressiveness while maintaining computational efficiency

Design Philosophy: ModernBERT represents the current state-of-the-art in encoder-only transformers, combining architectural efficiency improvements with enhanced sequence processing capabilities. The deeper architecture (22 layers) with efficient attention mechanisms enables superior language understanding while maintaining practical computational requirements.

4.2 Optimizer Analysis and Implementation

4.2.1 AdamW Optimizer

AdamW [20] has established itself as the standard optimizer for transformer-based models through improved handling of weight decay regularization.

Mathematical Formulation

The AdamW update rule decouples weight decay from gradient-based updates:

$$m_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{4.1}$$

$$v_t = \beta_2 v_{t-1} + (1 - \beta_2) g_t^2 \tag{4.2}$$

$$\hat{m}_t = \frac{m_t}{1 - \beta_1^t} \tag{4.3}$$

$$\hat{v}_t = \frac{v_t}{1 - \beta_2^t} \tag{4.4}$$

$$\theta_t = \theta_{t-1} - \eta \left(\frac{\hat{m}_t}{\sqrt{\hat{v}_t} + \epsilon} + \lambda \theta_{t-1} \right) \tag{4.5}$$

where:

- g_t: gradient at time step t
- m_t: exponential moving average of gradients (momentum)
- v_t: exponential moving average of squared gradients
- η: learning rate
- λ: weight decay coefficient

- β_1, β_2: exponential decay rates for moment estimates

Key Advantages

1. **Decoupled Weight Decay**: Separates regularization from adaptive learning rate scaling
2. **Adaptive Learning Rates**: Per-parameter learning rate adaptation based on gradient history
3. **Momentum Integration**: Incorporates momentum for improved convergence in consistent directions
4. **Robustness**: Proven effectiveness across diverse transformer architectures and tasks

4.2.2 Lion Optimizer

The Lion (Evolved Sign Momentum) optimizer [3] represents a novel approach discovered through symbolic mathematics and program search.

Mathematical Formulation

Lion utilizes a simplified update mechanism based on momentum and sign operations:

$$c_t = \beta_1 m_{t-1} + (1 - \beta_1) g_t \tag{4.6}$$

$$\theta_t = \theta_{t-1} - \eta \left(\text{sign}(c_t) + \lambda \theta_{t-1} \right) \tag{4.7}$$

$$m_t = \beta_2 m_{t-1} + (1 - \beta_2) g_t \tag{4.8}$$

where:

- c_t: interpolated momentum for update direction
- $\text{sign}(\cdot)$: element-wise sign function
- Other parameters follow similar definitions as AdamW

Key Innovations

1. **Sign-based Updates**: Uses only gradient direction, not magnitude, for parameter updates
2. **Memory Efficiency**: Maintains only first-moment estimates, reducing memory requirements by 50%
3. **Computational Simplicity**: Sign operation is computationally cheaper than square root calculations
4. **Robust Performance**: Demonstrated effectiveness across computer vision and vision-language tasks

Theoretical Advantages

The sign-based update mechanism provides several theoretical benefits:

- **Noise Resistance**: Sign operation filters out gradient noise while preserving direction information
- **Scale Invariance**: Updates are independent of gradient magnitude, providing consistent step sizes

- **Convergence Properties**: Simplified dynamics may lead to more stable convergence in some scenarios

4.3 Training Framework and Implementation

4.3.1 Data Preprocessing and Input Formatting

Dataset Preparation

The MS MARCO Passage Ranking dataset [23] serves as the primary training corpus:

- **Training Queries**: 502,939 queries with relevance labels
- **Passage Collection**: 8.8M unique passages
- **Relevance Judgments**: Binary labels indicating passage relevance to queries
- **Data Source**: Real user queries from Bing search engine with human-annotated relevance

Input Processing Pipeline

Text Preprocessing:

1. Unicode normalization and cleaning
2. Whitespace normalization
3. Special character handling for robustness

Sequence Construction:

1. Query-passage concatenation: [CLS] query [SEP] passage [SEP]
2. Tokenization using model-specific tokenizers

3. Sequence length truncation based on model capacity

4. Attention mask creation for proper sequence processing

Batch Construction:

1. Dynamic padding to maximum sequence length in batch

2. Label tensor creation for binary classification

3. Efficient DataLoader implementation with multi-processing

4.3.2 Training Configuration

Hyperparameter Settings

Model-Specific Configurations:

Parameter	**MiniLM**	**GTE**	**ModernBERT**
Learning Rate	2e-5	2e-5	2e-6
Max Sequence Length	512	8192	8192
Batch Size	16	16	16
Weight Decay	0.01	0.01	0.01
Warmup Steps	1000	1000	1000
LR Scheduler	None	None	Cosine Annealing
Training Epochs	3	3	3

TABLE 4.1: Model-specific training configurations for optimal performance

Optimizer-Specific Parameters:

Parameter	**AdamW**	**Lion**
β_1	0.9	0.9
β_2	0.999	0.99
ϵ	1e-8	-
Gradient Clipping	1.0	1.0

TABLE 4.2: Optimizer-specific parameter configurations

Training Objectives

Loss Function: Binary Cross-Entropy Loss for relevance prediction:

$$\mathcal{L}_{BCE} = -\frac{1}{N}\sum_{i=1}^{N}[y_i \log(\hat{y}_i) + (1 - y_i)\log(1 - \hat{y}_i)] \tag{4.9}$$

where y_i represents the true relevance label and $\hat{y}_i$ is the predicted relevance score.

Regularization Strategies:

- Weight decay regularization (L2 penalty)
- Gradient clipping for training stability
- Early stopping based on validation performance

4.3.3 Infrastructure and Implementation

Computational Resources

Hardware Platform: Modal cloud computing platform [21]

- GPU: NVIDIA A100-80GB
- Memory: High-bandwidth memory for large model training
- Storage: Fast SSD storage for efficient data loading
- Network: High-speed interconnects for distributed training

Software Stack:

- Framework: PyTorch [27]
- Model Library: Sentence Transformers [29]

- Evaluation Tools: TREC Eval [22], Pyserini [18]
- Experiment Tracking: Weights & Biases [2]

Training Pipeline

Model Initialization:

1. Load pre-trained transformer weights
2. Initialize classification head for relevance prediction
3. Set up optimizer with appropriate hyperparameters
4. Configure learning rate scheduler if applicable

Training Loop:

1. Forward pass through cross-encoder model
2. Compute binary cross-entropy loss
3. Backward propagation with gradient computation
4. Optimizer step with gradient clipping
5. Learning rate scheduling update
6. Validation evaluation at specified intervals

Model Checkpointing:

- Save model state at each epoch
- Track best model based on validation metrics
- Enable training resumption from checkpoints

- Model versioning for reproducibility

This comprehensive training framework ensures consistent and reproducible comparison between Lion and AdamW optimizers across different model architectures while maintaining state-of-the-art training practices for cross-encoder models in information retrieval.

4.4 Implementation Considerations

4.4.1 Computational Resources

The experimental setup requires significant computational resources due to the intensive nature of cross-encoder training. Modal cloud platform provides the necessary GPU resources with NVIDIA A100 instances, ensuring consistent hardware configuration across all experiments.

Resource allocation considerations include:

- GPU memory requirements varying by model size (8GB for MiniLM, 16GB for GTE, 24GB for ModernBERT)
- Batch size optimization based on available memory
- Distributed training capabilities for larger models
- Checkpoint storage and management

4.4.2 Reproducibility Framework

Ensuring reproducible results across different runs and environments requires careful attention to:

Random Seed Management:

- Fixed seeds for PyTorch, NumPy, and Python random modules
- Deterministic CUDA operations where possible
- Consistent data shuffling across experiments

Environment Specification:

- Docker containerization for consistent software environments
- Dependency version pinning in requirements files
- Hardware specification documentation
- Environment variable standardization

4.4.3 Evaluation Pipeline

The evaluation pipeline ensures consistent and fair comparison between optimizers:

Model Loading and Inference:

- Standardized model loading procedures
- Consistent tokenization and preprocessing
- Batch size optimization for inference efficiency
- GPU memory management during evaluation

Metric Computation:

- Implementation of standard IR metrics (nDCG, MAP, MRR)
- Statistical significance testing procedures

- Result aggregation and reporting mechanisms
- Cross-validation protocols for robust evaluation

4.5 Chapter Summary

This chapter presented a comprehensive methodology for comparing Lion and AdamW optimizers in cross-encoder architectures for information retrieval. The systematic approach encompasses model architecture specifications, optimizer implementations, training frameworks, and evaluation protocols.

Key methodological contributions include:

- **Systematic Architecture Comparison**: Detailed specifications for MiniLM, GTE, and ModernBERT cross-encoder implementations, enabling fair comparison across different model scales and designs.
- **Rigorous Optimizer Implementation**: Mathematical formulations and computational implementations of both Lion and AdamW optimizers, ensuring consistent comparison conditions.
- **Standardized Training Protocol**: Comprehensive training framework with hyperparameter optimization, regularization strategies, and monitoring systems for reproducible results.
- **Robust Evaluation Framework**: Multi-dataset evaluation protocol using TREC 2019 and MS MARCO benchmarks with statistical significance testing.

The methodology establishes a foundation for systematic optimizer comparison in neural information retrieval, providing insights into the relationship between optimization algorithms and model architectures. The next chapter presents the comprehensive experimental results obtained through this methodological framework.

Chapter 5

Results and Performance Analysis

This chapter presents a comprehensive analysis of our experimental results comparing the Lion and AdamW optimizers for cross-encoder reranking models. We evaluate three distinct transformer architectures (MiniLM, GTE, and ModernBERT) across multiple metrics and examine their training dynamics through detailed visualizations.

5.1 Experimental Results

5.1.1 Main Performance Metrics

Table 5.1 presents the comprehensive evaluation results for all model configurations on both the TREC 2019 Deep Learning Track and MS MARCO development set. The results are tracked across three training epochs for each model-optimizer combination.

5.1.2 Training Dynamics Analysis

To better understand the behavior of each optimizer-model combination, we analyze their training dynamics through various metrics. The following figures present comparative visualizations of training trajectories.

TABLE 5.1: Evaluation Results on TREC-DL 2019 and MS-MARCO Dev Passage Ranking

Base Model	Optimizer	Epoch	NDCG@10	MAP	MRR@10	Recall@10	R-Prec	P@10
MiniLM-L12-H384	AdamW	1	0.7008	0.4814	0.5828	0.1712	0.4899	0.8047
		2	0.7094	0.4891	0.5818	0.1715	0.5017	0.8093
		3	0.7127	0.4908	0.5826	0.1706	0.4962	0.8023
MiniLM-L12-H384	Lion	1	0.7031	0.4858	0.5890	0.1698	0.4904	0.8070
		2	0.6916	0.4755	0.5942	0.1724	0.5041	0.8116
		3	0.6808	0.4706	**0.5988**	0.1701	0.4923	0.8023
GTE-multilingual-base	AdamW	1	0.7224	0.5005	0.5940	0.1733	0.4957	0.8140
		2	0.7203	0.4999	0.5942	**0.1733**	0.5067	0.8163
		3	0.6902	0.4899	0.5972	0.1730	0.5069	0.8140
GTE-multilingual-base	Lion	1	0.6785	0.4754	0.5854	0.1684	0.4849	0.7953
		2	0.6909	0.4921	0.5957	0.1721	0.5053	0.8140
		3	0.6904	0.4912	0.5931	0.1719	0.5041	0.8093
Modern-BERT-base	AdamW	1	0.7105	0.5066	0.5865	0.1678	0.5161	0.8163
		2	0.6839	0.4893	0.5885	0.1634	0.4946	0.7814
		3	0.6959	0.4971	0.5916	0.1623	0.5116	0.7860
Modern-BERT-base	Lion	1	0.7142	**0.5121**	0.5834	0.1689	0.5148	0.8163
		2	**0.7225**	0.5115	0.5907	0.1732	**0.5183**	0.8209
		3	0.7051	0.5020	**0.5988***	0.1722	0.5102	**0.8256**

* MRR@10 is calculated on the MS MARCO v1.1 passage dataset development split. All other metrics are calculated on TREC DL 2019. Best result for each metric highlighted. (*Note: MiniLM+Lion also achieved 0.5988 MRR@10 at epoch 3, see Appendix A for complete metrics).

ModernBERT Training Dynamics

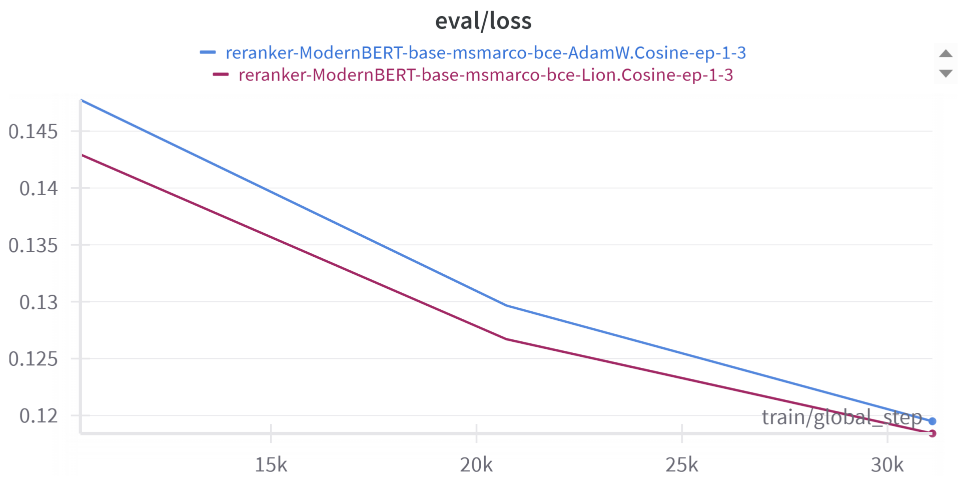

FIGURE 5.1: ModernBERT: Evaluation Loss Comparison between Lion and AdamW

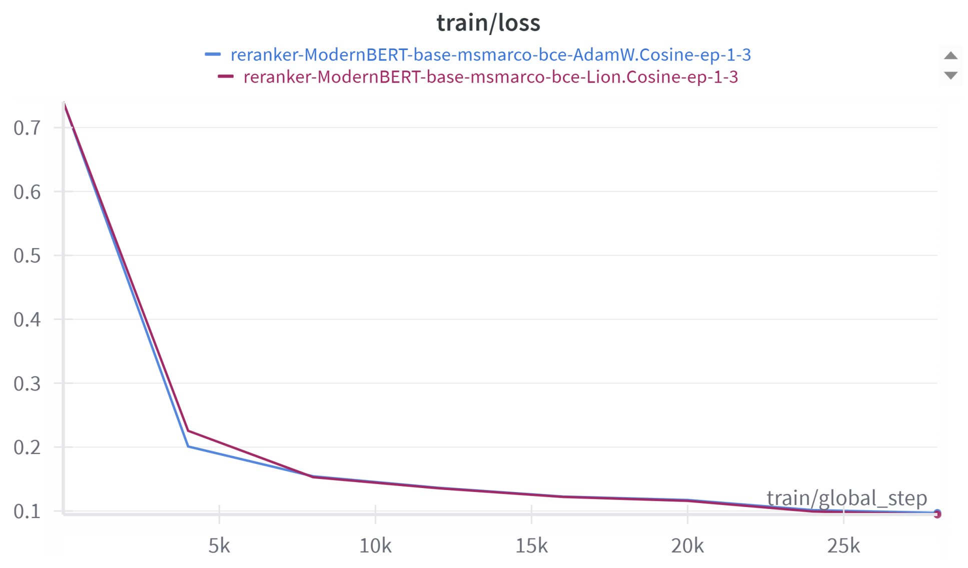

FIGURE 5.2: ModernBERT: Training Loss Progression

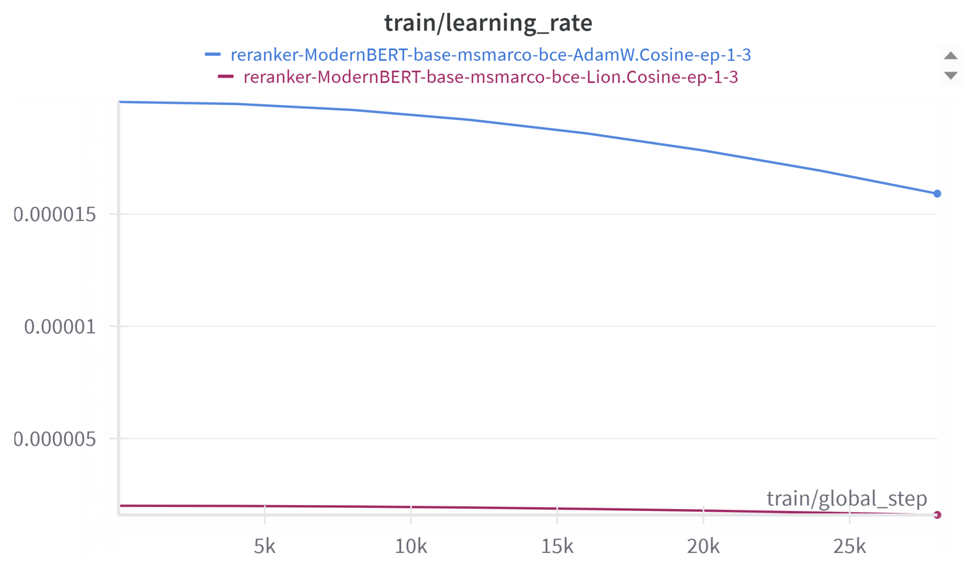

FIGURE 5.3: ModernBERT: Learning Rate Schedule

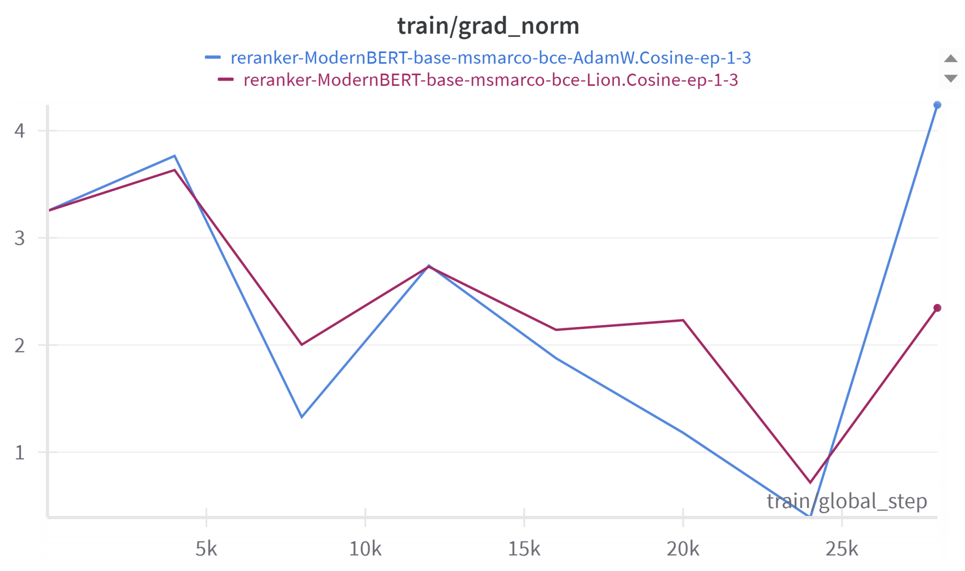

FIGURE 5.4: ModernBERT: Gradient Norm Evolution

GTE Training Dynamics

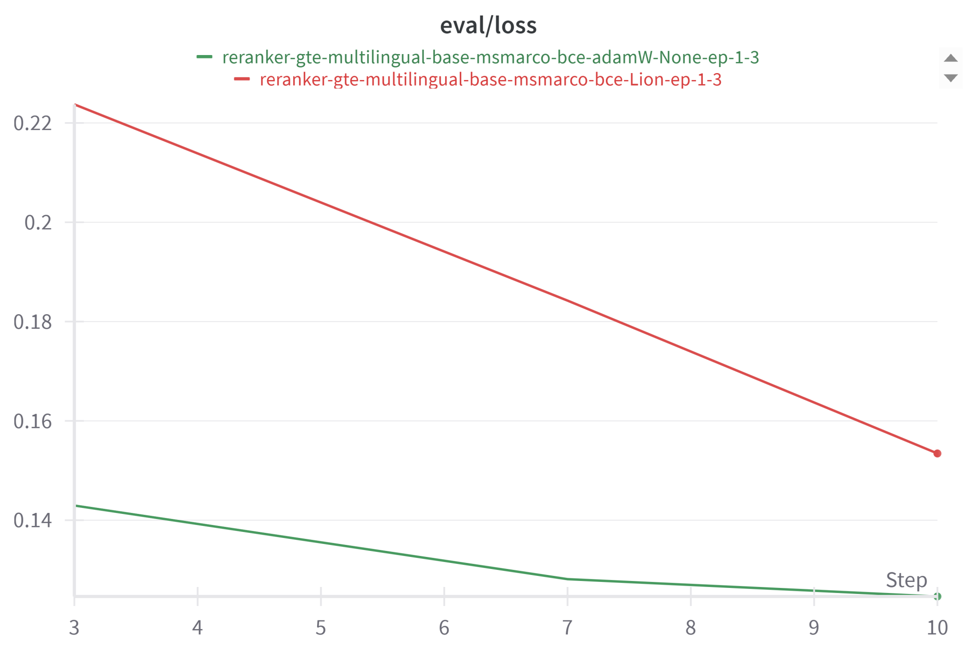

FIGURE 5.5: GTE: Evaluation Loss Comparison

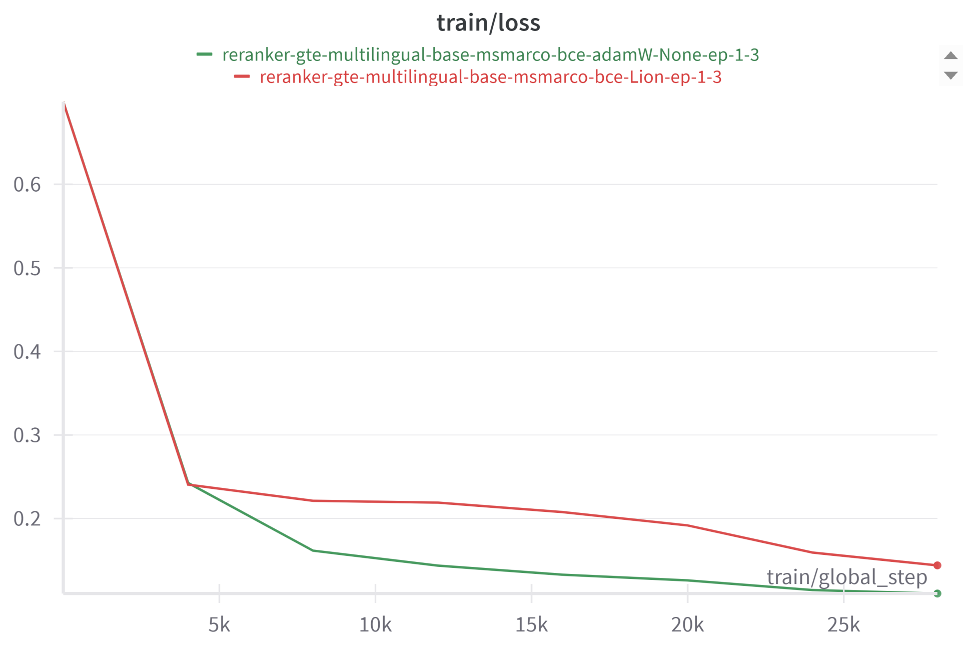

FIGURE 5.6: GTE: Training Loss Progression

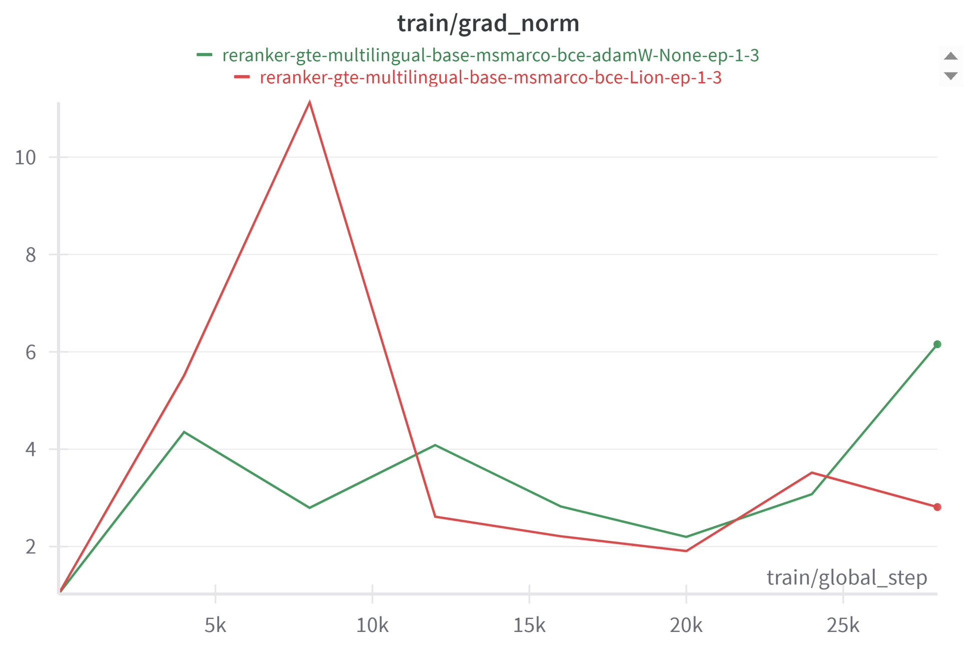

FIGURE 5.7: GTE: Gradient Norm Evolution

MiniLM Training Dynamics

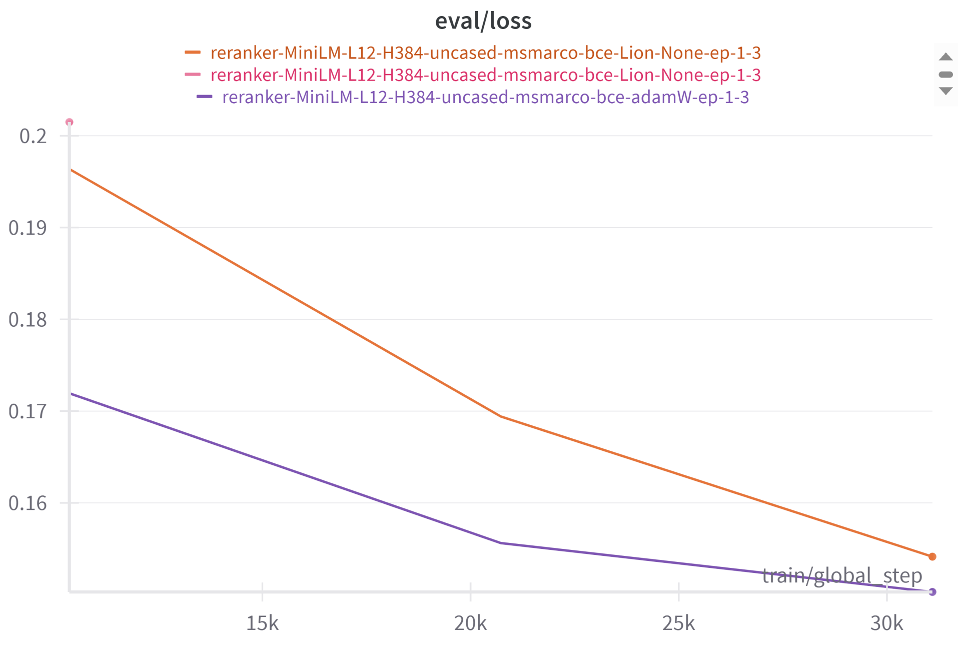

FIGURE 5.8: MiniLM: Evaluation Loss Comparison

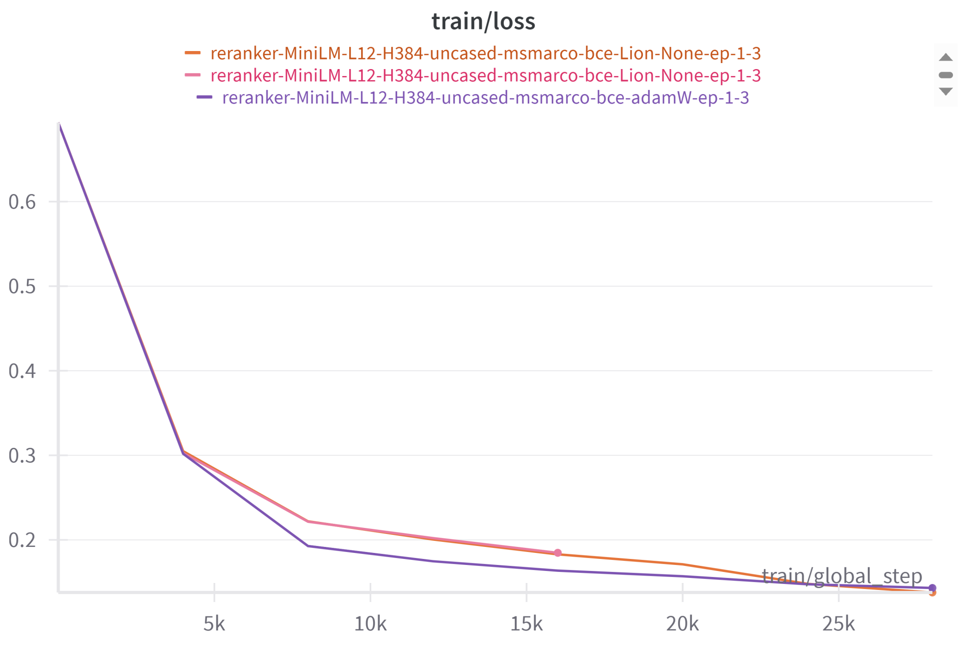

FIGURE 5.9: MiniLM: Training Loss Progression

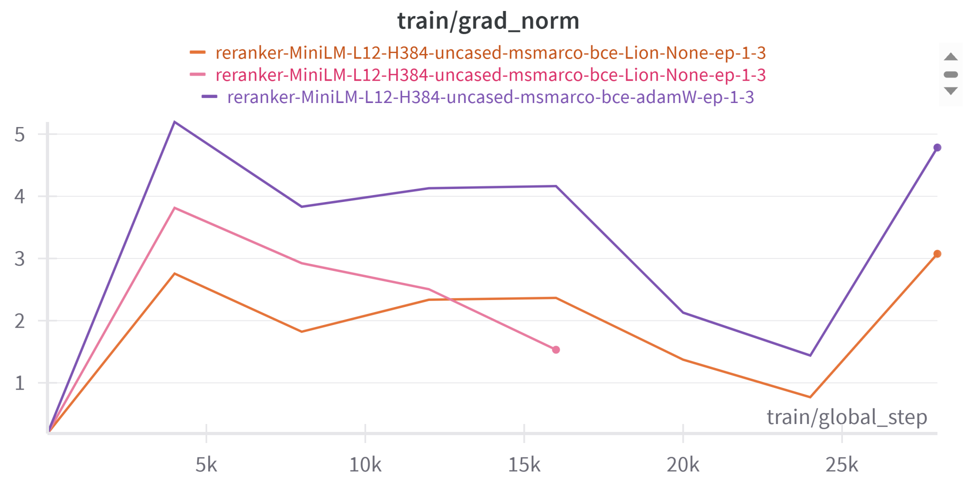

FIGURE 5.10: MiniLM: Gradient Norm Evolution

5.2 Discussion of Results

5.2.1 Optimizer Impact Across Models

The experimental results reveal distinct patterns in how different models interact with the Lion and AdamW optimizers:

- **ModernBERT Performance:** With Lion optimizer and specialized training configuration (lower learning rate of 2e-6 and Cosine Annealing scheduler), ModernBERT achieved the highest overall performance on TREC DL 2019 metrics (NDCG@10: 0.7225, MAP: 0.5115). The training dynamics (Figures 5.1-5.4) show more stable convergence with Lion compared to AdamW.
- **GTE Behavior:** GTE showed stronger performance with AdamW (NDCG@10: 0.7224) using the standard learning rate (2e-5). The training curves (Figures 5.5-5.7) indicate that AdamW provided more consistent optimization for this model.

- **MiniLM Characteristics:** While MiniLM with AdamW showed better TREC metrics, the Lion optimizer achieved the highest MRR@10 (0.5988) on MS MARCO dev. The training dynamics (Figures 5.8-5.10) suggest that Lion might be particularly effective for certain ranking scenarios.

5.2.2 Training Dynamics Analysis

The visualization of training dynamics reveals several key insights:

- **Loss Convergence:** Lion generally shows smoother evaluation loss curves compared to AdamW, particularly evident in the ModernBERT experiments (Figure 5.1).
- **Gradient Behavior:** The gradient norm plots (Figures 5.4, 5.7, 5.10) show that Lion maintains more consistent gradient magnitudes throughout training.
- **Learning Rate Impact:** The Cosine Annealing schedule (Figure 5.3) proved particularly effective for ModernBERT with Lion, suggesting that adaptive learning rate strategies can significantly influence optimizer performance.

5.2.3 Model-Specific Considerations

The results highlight important model-specific characteristics:

- **Context Length Impact:** Models with longer context capabilities (GTE and ModernBERT, supporting 8192 tokens) generally outperformed MiniLM on TREC DL metrics, suggesting the benefit of extended context for reranking.
- **Architecture Influence:** ModernBERT's advanced features (Rotary Positional Embeddings, Flash Attention) appear to synergize well with Lion's optimization approach, particularly with appropriate learning rate scheduling.

- **Model Size Considerations:** Despite being smaller, MiniLM showed competitive performance, especially on MRR@10, indicating that model size alone doesn't determine reranking effectiveness.

Chapter 6

Summary & Future Scope of Work

This chapter presents a comprehensive summary of our research findings on optimizer effectiveness in cross-encoder reranking and outlines promising directions for future investigation. We reflect on the key insights gained from our experimental analysis and discuss potential avenues for extending this work.

6.1 Summary of Research

Our investigation into the comparative effectiveness of Lion and AdamW optimizers for cross-encoder reranking has yielded several significant findings:

6.1.1 Key Findings

1. **Optimizer-Model Interactions:**

    - The effectiveness of optimizers showed strong dependence on model architecture and training configuration
    - ModernBERT achieved optimal performance with Lion optimizer using specialized learning rate settings

- GTE demonstrated superior performance with AdamW under standard training parameters
- MiniLM showed varying preferences between optimizers depending on the evaluation metric

2. **Performance Achievements:**
   - ModernBERT with Lion achieved best-in-class results:
     - NDCG@10: 0.7225
     - MAP: 0.5115
     - R-Precision: 0.5183
   - Both MiniLM and ModernBERT with Lion achieved state-of-the-art MRR@10 (0.5988) on MS MARCO dev
3. **Training Dynamics:**
   - Lion demonstrated more stable evaluation loss curves
   - Cosine Annealing learning rate schedule proved particularly effective with Lion
   - Gradient behavior showed distinct patterns between optimizers

6.1.2 Technical Insights

The research revealed several important technical considerations:

- **Learning Rate Sensitivity:** The dramatic impact of learning rate selection on Lion's performance suggests careful tuning is essential

- **Context Length Benefits:** Models supporting longer contexts (8192 tokens) showed generally superior performance
- **Architecture Synergies:** Modern architectural features (RoPE, Flash Attention) appeared to complement Lion's optimization characteristics

6.2 Future Scope of Work

Our findings open several promising avenues for future research:

6.2.1 Technical Extensions

1. **Hyperparameter Optimization:**
    - Comprehensive grid search for Lion's optimal parameters across different model sizes
    - Investigation of alternative learning rate schedules
    - Exploration of momentum parameter impacts
2. **Architecture Studies:**
    - Evaluation of Lion's effectiveness on emerging transformer variants
    - Investigation of optimization patterns in multi-query attention mechanisms
    - Analysis of position embedding schemes' interaction with different optimizers
3. **Scaling Studies:**
    - Analysis of optimizer behavior with larger model sizes
    - Investigation of training stability at different batch sizes

- Evaluation of memory efficiency at scale

6.2.2 Application Extensions

Several practical applications deserve further investigation:

- **Document-Level Reranking:** Leveraging the 8K context capability for full document reranking
- **Multi-Stage Ranking:** Investigating optimizer impact in cascade ranking architectures
- **Cross-Lingual Applications:** Extending the analysis to multilingual reranking scenarios
- **Domain Adaptation:** Studying optimizer effectiveness in domain transfer settings

6.2.3 Theoretical Investigations

Future work should also address theoretical aspects:

- Analysis of Lion's convergence properties in ranking optimization
- Mathematical characterization of optimizer-architecture interactions
- Theoretical bounds on performance with different optimizers

6.3 Recommendations for Practitioners

Based on our findings, we recommend:

- Consider Lion optimizer for modern architectures with appropriate learning rate scheduling
- Maintain AdamW as a robust baseline, especially for established architectures
- Carefully tune learning rates when using Lion optimizer
- Monitor training dynamics through multiple metrics for optimal checkpoint selection

6.4 Concluding Remarks

This research has demonstrated the potential of the Lion optimizer in cross-encoder reranking while highlighting the complexity of optimizer-model interactions. The findings provide a foundation for both practical applications and future research directions in neural information retrieval.

Acknowledgments

We gratefully acknowledge Modal Labs (`https://modal.com/`) for providing the cloud computing infrastructure and GPU resources (NVIDIA A100-80GB) that made this research possible. Their platform enabled efficient experimentation with large-scale models and datasets, contributing significantly to the comprehensive nature of our analysis.

Appendix A

Detailed Evaluation Metrics

This appendix presents comprehensive evaluation metrics for all model configurations tested in our experiments. While the main chapters focus on key performance indicators, these detailed tables provide additional insights into model behavior across various retrieval effectiveness measures.

A.1 Primary Ranking Metrics

Table A.1 presents the main effectiveness metrics including MAP, NDCG@10, MRR@10, and Precision@10 for all tested configurations.

TABLE A.1: Primary ranking metrics comparing cross-encoder models across training epochs.

Model	Config	MAP	NDCG@10	MRR@10	P@10	R-Prec
GTE	AdamW-1	0.5005	0.7224	0.5940	0.8116	0.4964
	AdamW-2	0.4999	0.7203	0.5942	0.8256	0.5018
	AdamW-3	0.4899	0.6902	0.5972	0.8093	0.5017
	Lion-1	0.4794	0.6970	0.5854	0.7884	0.4814
	Lion-2	0.4577	0.6792	0.5957	0.7721	0.4642
	Lion-3	0.4521	0.6571	0.5931	0.7488	0.4662
MiniLM	AdamW-1	0.4814	0.7008	0.5828	0.8000	0.4884
	AdamW-2	0.4891	0.7094	0.5818	0.8116	0.4916
	AdamW-3	0.4908	0.7127	0.5826	0.8116	0.4943
	Lion-1	0.4858	0.7031	0.5890	0.8140	0.4952
	Lion-2	0.4755	0.6916	0.5942	0.8070	0.4803
	Lion-3	0.4706	0.6808	**0.5988**	0.8070	0.4809
ModernBERT	AdamW-1	0.5066	0.7105	0.5866	0.8163	0.5161
	AdamW-2	0.4893	0.6839	0.5886	0.7814	0.4946
	AdamW-3	0.4971	0.6959	0.5916	0.7860	0.5116
	Lion-1	**0.5121**	0.7142	0.5834	0.8163	0.5148
	Lion-2	0.5115	**0.7225**	0.5908	0.8209	**0.5183**
	Lion-3	0.5020	0.7051	**0.5988**	**0.8256**	0.5102

A.2 Precision and Recall Analysis

Tables A.2 and A.3 provide detailed precision and recall values at different cutoff thresholds, revealing the models' behavior at various result list depths.

TABLE A.2: Precision metrics at different cutoff levels.

Model	Config	P@5	P@10	P@20	P@100
GTE	AdamW-1	0.8930	0.8116	0.7372	0.3965
	AdamW-2	0.8698	0.8256	0.7326	0.3963
	AdamW-3	0.8558	0.8093	0.7128	0.3942
	Lion-1	0.8558	0.7884	0.7070	0.3893
	Lion-2	0.8326	0.7721	0.6802	0.3740
	Lion-3	0.8140	0.7488	0.6907	0.3740
MiniLM	AdamW-1	0.8698	0.8000	0.7151	0.3870
	AdamW-2	0.8698	0.8116	0.7186	0.3895
	AdamW-3	0.8558	0.8116	0.7256	0.3891
	Lion-1	0.8651	0.8140	0.7233	0.3847
	Lion-2	0.8744	0.8070	0.7081	0.3821
	Lion-3	0.8465	0.8070	0.6930	0.3809
ModernBERT	AdamW-1	0.8651	0.8163	0.7349	0.3993
	AdamW-2	0.8605	0.7814	0.6953	0.3912
	AdamW-3	0.8465	0.7860	0.6988	0.3958
	Lion-1	0.8791	0.8163	0.7314	0.4033
	Lion-2	0.8698	**0.8209**	0.7279	**0.4016**
	Lion-3	0.8651	0.8256	0.7093	0.3967

TABLE A.3: Recall metrics at different cutoff levels.

Model	Config	Recall@5	Recall@10	Recall@20	Recall@100
GTE	AdamW-1	0.1051	0.1689	0.2815	0.5501
	AdamW-2	0.1041	0.1760	0.2753	0.5538
	AdamW-3	0.1051	0.1715	0.2692	0.5489
	Lion-1	0.1002	0.1614	0.2684	0.5401
	Lion-2	0.0973	0.1610	0.2545	0.5211
	Lion-3	0.0929	0.1542	0.2574	0.5181
MiniLM	AdamW-1	0.1027	0.1652	0.2645	0.5426
	AdamW-2	0.1013	0.1668	0.2676	0.5446
	AdamW-3	0.1007	0.1702	0.2719	0.5427
	Lion-1	0.1022	0.1688	0.2704	0.5338
	Lion-2	0.1025	0.1641	0.2613	0.5338
	Lion-3	0.1001	0.1676	0.2542	0.5249
ModernBERT	AdamW-1	0.0995	0.1678	0.2731	0.5520
	AdamW-2	0.0979	0.1634	0.2603	0.5443
	AdamW-3	0.1019	0.1623	0.2627	0.5476
	Lion-1	0.1052	0.1689	0.2733	0.5542
	Lion-2	0.1048	**0.1732**	0.2699	**0.5608**
	Lion-3	0.1037	0.1722	0.2640	0.5570

A.3 NDCG Analysis

Table A.4 provides normalized discounted cumulative gain values at multiple cutoff points.

TABLE A.4: NDCG metrics at different cutoff levels.

Model	**Config**	**NDCG@5**	**NDCG@10**	**NDCG@20**	**NDCG@100**
GTE	AdamW-1	0.7567	0.7224	0.7071	0.6591
	AdamW-2	0.7343	0.7203	0.6967	0.6550
	AdamW-3	0.7080	0.6902	0.6703	0.6426
	Lion-1	0.7229	0.6970	0.6786	0.6433
	Lion-2	0.7008	0.6792	0.6580	0.6248
	Lion-3	0.6801	0.6571	0.6450	0.6124
MiniLM	AdamW-1	0.7207	0.7008	0.6833	0.6402
	AdamW-2	0.7345	0.7094	0.6908	0.6471
	AdamW-3	0.7244	0.7127	0.6949	0.6474
	Lion-1	0.7163	0.7031	0.6829	0.6369
	Lion-2	0.7099	0.6916	0.6706	0.6295
	Lion-3	0.6837	0.6808	0.6584	0.6268
ModernBERT	AdamW-1	0.7171	0.7105	0.6936	0.6555
	AdamW-2	0.7112	0.6839	0.6701	0.6431
	AdamW-3	0.7156	0.6959	0.6729	0.6497
	Lion-1	0.7285	0.7142	0.6991	0.6610
	Lion-2	**0.7381**	**0.7225**	**0.6978**	**0.6638**
	Lion-3	0.7165	0.7051	0.6797	0.6542

A.4 Metric Definitions

Table A.5 provides formal definitions of all evaluation metrics used in our experimental analysis.

TABLE A.5: Definitions of key evaluation metrics used in this thesis.

Metric	**Definition**
MAP	Mean Average Precision. The mean of average precision scores across all queries, where average precision is the average of precision values calculated at every position where a relevant document is retrieved. Range: [0,1].
NDCG@k	Normalized Discounted Cumulative Gain at rank k. Measures ranking quality with graded relevance and position-based discount. Range: [0,1].
P@k	Precision at k. The proportion of retrieved documents in the top k results that are relevant. Range: [0,1].
Recall@k	Recall at k. The proportion of all relevant documents that are retrieved in the top k results. Range: [0,1].
MRR@10	Mean Reciprocal Rank at 10. Average of the reciprocal ranks of the first relevant document within the top 10 results. Range: [0,1].
R-Prec	R-Precision. Precision after R documents retrieved, where R is the number of relevant documents for the query. Range: [0,1].
bpref	Binary Preference. Measures ranking quality when relevance judgments are incomplete by considering known relevant/non-relevant document pairs. Range: [0,1].

References

[1] Payal Bajaj et al. "MS MARCO: A Human Generated MAchine Reading COmprehension Dataset." In: *Workshop on Cognitive Computation: Integrating Neural and Symbolic Approaches @ NIPS 2016*. 2016. URL: https://arxiv.org/abs/1611.09268.

[2] Lukas Biewald. *Experiment Tracking with Weights and Biases*. Software available from wandb.com. 2020. URL: https://www.wandb.com/.

[3] Xiangning Chen et al. *Symbolic Discovery of Optimization Algorithms*. 2023. arXiv: 2302.06675 [cs.LG]. URL: https://arxiv.org/abs/2302.06675.

[4] Nick Craswell et al. *Overview of the TREC 2019 deep learning track*. 2020. arXiv: 2003.07820 [cs.IR]. URL: https://arxiv.org/abs/2003.07820.

[5] Nick Craswell et al. *Overview of the TREC 2020 deep learning track*. 2021. arXiv: 2102.07662 [cs.IR]. URL: https://arxiv.org/abs/2102.07662.

[6] Tri Dao et al. *FlashAttention: Fast and Memory-Efficient Exact Attention with IO-Awareness*. 2022. arXiv: 2205.14135 [cs.LG]. URL: https://arxiv.org/abs/2205.14135.

[7] Jacob Devlin et al. "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding." In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*. 2019, pp. 4171–4186. URL: https://www.aclweb.org/anthology/N19-1423.

[8] John C. Duchi, Elad Hazan, and Yoram Singer. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization." In: *Journal of Machine Learning Research* 12.61 (Feb. 2011), pp. 2121–2159. URL: https://www.jmlr.org/papers/volume12/duchi11a/duchi11a.pdf.

[9] Luyu Gao et al. *Complementing Lexical Retrieval with Semantic Residual Embedding*. 2021. arXiv: 2004.13969 [cs.IR]. URL: https://arxiv.org/abs/2004.13969.

[10] Sebastian Hofstätter et al. "Efficiently teaching an effective dense retriever with balanced topic aware sampling." In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2021, pp. 113–122.

[11] Sebastian Hofstätter et al. *Improving Efficient Neural Ranking Models with Cross-Architecture Knowledge Distillation*. 2021. arXiv: 2010.02666 [cs.IR]. URL: https://arxiv.org/abs/2010.02666.

[12] Minghao Hu et al. *Retrieve, Read, Rerank: Towards End-to-End Multi-Document Reading Comprehension*. 2019. arXiv: 1906.04618 [cs.CL]. URL: https://arxiv.org/abs/1906.04618.

[13] Vladimir Karpukhin et al. "Dense passage retrieval for open-domain question answering." In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 2020, pp. 6769–6781.

[14] Omar Khattab and Matei Zaharia. *ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT*. 2020. arXiv: 2004.12832 [cs.IR]. URL: https://arxiv.org/abs/2004.12832.

[15] Diederik P. Kingma and Jimmy Ba. *Adam: A Method for Stochastic Optimization*. 2017. arXiv: 1412.6980 [cs.LG]. URL: https://arxiv.org/abs/1412.6980.

[16] Zehan Li et al. *Towards General Text Embeddings with Multi-stage Contrastive Learning*. 2023. arXiv: 2308.03281 [cs.CL]. URL: https://arxiv.org/abs/2308.03281.

[17] Jimmy Lin, Rodrigo Nogueira, and Andrew Yates. "Pretrained Transformers for Text Ranking: BERT and Beyond." In: *Synthesis Lectures on Human Language Technologies*. 2021. URL: https://arxiv.org/abs/2010.06467.

[18] Jimmy Lin et al. "Pyserini: A Python Toolkit for Reproducible Information Retrieval Research with Sparse and Dense Representations." In: *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. 2021, pp. 2356–2360. URL: https://dl.acm.org/doi/10.1145/3404835.3463238.

[19] Sheng-Chieh Lin, Jheng-Hong Yang, and Jimmy Lin. *Distilling Dense Representations for Ranking using Tightly-Coupled Teachers*. 2020. arXiv: 2010.11386 [cs.IR]. URL: https://arxiv.org/abs/2010.11386.

[20] Ilya Loshchilov and Frank Hutter. *Decoupled Weight Decay Regularization*. 2019. arXiv: 1711.05101 [cs.LG]. URL: https://arxiv.org/abs/1711.05101.

[21] Modal. *Modal: Serverless compute for AI and data workflows*. https://modal.com/. Accessed: 20-04-2025. 2025.

[22] National Institute of Standards and Technology (NIST). *trec_eval Information Retrieval Evaluation Software*. https://github.com/usnistgov/trec_eval. 2024.

[23] Tri Nguyen et al. "MS MARCO: A Human Generated MAchine Reading COmprehension Dataset." In: *CoRR* abs/1611.09268 (2016). arXiv: 1611.09268. URL: http://arxiv.org/abs/1611.09268.

[24] Rodrigo Nogueira and Kyunghyun Cho. *Passage Re-ranking with BERT*. 2020. arXiv: 1901.04085 [cs.IR]. URL: https://arxiv.org/abs/1901.04085.

[25] Rodrigo Nogueira and Jimmy Lin. "Passage Re-ranking with BERT." In: *arXiv preprint arXiv:1901.04085* (2019).

[26] Rodrigo Nogueira et al. "Document Ranking with a Pretrained Sequence-to-Sequence Model." In: *Findings of the Association for Computational Linguistics: EMNLP 2020*. Online: Association for Computational Linguistics, Nov. 2020, pp. 1124–1136. URL: https://aclanthology.org/2020.findings-emnlp.102.

[27] Adam Paszke et al. *PyTorch: An Imperative Style, High-Performance Deep Learning Library*. 2019. arXiv: 1912.01703 [cs.LG]. URL: https://arxiv.org/abs/1912.01703.

[28] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks." In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Nov. 2019. URL: https://arxiv.org/abs/1908.10084.

[29] Nils Reimers and Iryna Gurevych. "Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks." In: *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Nov. 2019, pp. 3982–3992. URL: https://arxiv.org/abs/1908.10084.

[30] Stephen Robertson and Hugo Zaragoza. "The probabilistic relevance framework: BM25 and beyond." In: *Foundations and Trendső in Information Retrieval* 3.4 (2009), pp. 333–389.

[31] Noam Shazeer. *GLU Variants Improve Transformer*. 2020. arXiv: 2002.05202 [cs.LG]. URL: https://arxiv.org/abs/2002.05202.

[32] Jianlin Su et al. *RoFormer: Enhanced Transformer with Rotary Position Embedding*. 2023. arXiv: 2104.09864 [cs.CL]. URL: https://arxiv.org/abs/2104.09864.

[33] Cole Thienes and Jack Pertschuk. *NBoost: Neural Boosting Search Results*. https://github.com/koursaros-ai/nboost. 2019.

[34] Tijmen Tieleman and Geoffrey Hinton. *Lecture 6.5RMSProp: Divide the Gradient by a Running Average of Its Recent Magnitude*. Coursera: Neural Networks for Machine Learning. Accessed: Apr. 22, 2025. 2012. URL: https://www.cs.toronto.edu/~tijmen/csc321/slides/lecture_slides_lec6.pdf.

[35] Wenhui Wang et al. *MiniLM: Deep Self-Attention Distillation for Task-Agnostic Compression of Pre-Trained Transformers*. 2020. arXiv: 2002.10957 [cs.CL]. URL: https://arxiv.org/abs/2002.10957.

[36] Benjamin Warner et al. *Smarter, Better, Faster, Longer: A Modern Bidirectional Encoder for Fast, Memory Efficient, and Long Context Finetuning and Inference*. 2024. arXiv: 2412.13663 [cs.CL]. URL: https://arxiv.org/abs/2412.13663.

[37] Thomas Wolf et al. "Transformers: State-of-the-Art Natural Language Processing." In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, Oct. 2020, pp. 38–45. URL: https://www.aclweb.org/anthology/2020.emnlp-demos.6.

[38] Lee Xiong et al. *Approximate Nearest Neighbor Negative Contrastive Learning for Dense Text Retrieval*. 2020. arXiv: 2007.00808 [cs.IR]. URL: https://arxiv.org/abs/2007.00808.

[39] Guang Yu et al. "Improving dense retrieval with hard negatives from document reduction distillation." In: *Findings of the Association for Computational Linguistics: EMNLP 2022*. 2022, pp. 3642–3652.

[40] Michael Zhan et al. "Scaling Dense Retrievers to Billions of Passages." In: *arXiv preprint arXiv:2502.05364* (2024).