University of Padova

Department of Information Engineering - DEI

Master Thesis in ICT for Internet and Multimedia

Evaluating Prefix-Driven Contrastive Learning for Semantic Search in High-Impact Societal Domains

Supervisor
Leonardo Badia
University of Padova

Co-supervisor
Antonio Lanza
Davide Costa

Master Candidate
Ceren Yilmaz

Student ID
2080637

Academic Year
2024-2025

Graduation Date
3 April 2025

I want to express my deepest gratitude to my beloved husband, whose unwavering support and unconditional love have been the foundation of my strength throughout both my bachelor's and master's journey. We studied together, side by side, helping and encouraging each other through every challenge, and I am endlessly grateful for his presence in my life.

To my mother and father, thank you for always being there for me with open hearts, providing guidance, love, and endless support—no matter the circumstances. You have shown me what it means to love without limits, and I can never thank you enough for everything you've done, including supporting my decision to pursue my master's abroad.

To my sister, your constant love and encouragement have meant the world to me, and I am so grateful for the bond we share that has only grown stronger over time.

I would also like to extend my heartfelt thanks to my company supervisors, Antonio Lanza and Davide Costa, from Altilia. Writing this thesis under your guidance has been an invaluable experience. Your mentorship has taught me so much, and I am truly grateful for all the knowledge and insights I have gained from you.

Abstract

Semantic search is a fundamental component of information retrieval, particularly in high-impact domains where extracting precise and contextually relevant information is crucial. This research investigates the effectiveness of **Prefix-Driven Contrastive Learning** in improving retrieval performance for **semantic search** by fine-tuning pretrained models. Specifically, the study focuses on **BGE (Better Generative Embeddings)** and **E5 (Embedding-based Efficiently-trained Encoder)**, leveraging contrastive learning techniques to enhance text representations.

The experiments are conducted using the **FiQA** dataset, a benchmark for financial question-answering and retrieval tasks, ensuring the evaluation is domain-specific and aligned with real-world applications. The study systematically applies **prefix-based training**, where corpus and query prefixes are explicitly incorporated to improve the alignment of embeddings. The impact of this approach is assessed using multiple retrieval models, including **BGE-small, BGE-base, and E5-small**. To ensure consistency, all models are fine-tuned using the same hyperparameters, with adjustments made only for **BGE-base** due to computational constraints.

The effectiveness of prefix-driven contrastive learning is evaluated through standard retrieval metrics, ensuring a comprehensive assessment of retrieval performance. The study employs a robust evaluation framework to analyze the impact of prefix-based training on model effectiveness in financial semantic search tasks.

The findings of this study contribute to **advancing semantic search methodologies** by analyzing the role of prefix-driven training in dense retrieval models. The results offer insights into the potential benefits and limitations of prefix-based contrastive learning, particularly for **financial and high-impact domains**. By identifying effective strategies for enhancing representation learning in retrieval systems, this research aims to inform future developments in **domain-specific search optimization**.

Contents

Listing of figures

Listing of tables

Listing of acronyms

IR . . . . . Information Retrieval

TF-IDF . . . . . Term Frequency-Inverse Document Frequency

ML . . . . . Machine Learning

DL . . . . . Deep Learning

MRR . . . . . Mean Reciprocal Rank

NDCG . . . . . Normalized Cumulative Gain

MAP . . . . . Mean Average Precision

CTR . . . . . Click-Through Rate

SVM . . . . . Support Vector Machine

KNN . . . . . k-Nearest Neighbors

CNN . . . . . Convolutional Neural Network

RNN . . . . . Recurrent Neural Network

BERT . . . . . Bidirectional Encoder Representations from Transformers

NLP . . . . . Natural Language Processing

Seq2Seq . . . . . Sequence-to-Sequence

HMM . . . . . Hidden Markov Models

SMT . . . . . Statistical Machine Translation

GRU . . . . . Gated Recurrent Unit

LSTM . . . . . Long Short-Term Memory

EOS . . . . . End of Sequence

FFN . . . . . The Position-Wise Feed-Forward Network

NER . . . . . Named Entity Recognition

NSP . . . . . Next Sentence Prediction

MLM . . . . . Masked Language Model

S-BERT . . . . . Sentence BERT

MNR . . . . . Multiple Negative Ranking Loss

RetroMAE . . . . . Retrospective Masked Autoencoders

1 Introduction

1.1 Information Retrieval: An Overview

In the era of digital transformation, the ability to efficiently access relevant information from massive amounts of unstructured data has become a fundamental necessity. **Information Retrieval (IR)** is a branch of computer science that focuses on the storage, organization, and retrieval of relevant information in response to user queries. [4] IR systems are designed to extract and rank relevant documents or data points based on their *relevance* to a given query, ensuring users receive accurate and useful results.

Historically, early IR systems relied on basic text-matching techniques such as **Boolean retrieval**, which used exact keyword matching to locate relevant documents.[4] This method, while effective for structured queries, struggled with *synonymy* (different words with the same meaning) and *polysemy* (words with multiple meanings), leading to both retrieval inefficiencies and poor ranking of results. Later, statistical techniques like **TF-IDF (Term Frequency-Inverse Document Frequency)** and **BM25** improved relevance ranking by weighing the importance of words within documents, allowing for more refined ranking mechanisms.[5]

The importance of IR is evident across various domains, including:

- **Search Engines:** Google, Bing, and DuckDuckGo rely on advanced IR techniques to rank and retrieve web pages efficiently.

- **Digital Libraries & Academic Research:** Platforms like Google Scholar and PubMed use IR to help researchers locate relevant scientific papers.
- **E-commerce & Recommendation Systems:** Platforms like Amazon, Netflix, and Spotify leverage IR models to provide personalized recommendations based on user preferences.
- **Healthcare & Biomedical Research:** IR is critical in retrieving medical records, clinical studies, and disease-related information. [6]
- **Legal & Financial Domains:** Legal professionals use IR for case law retrieval, while financial analysts rely on it for market trends and risk assessment.

The increasing reliance on **automated and intelligent search mechanisms** has driven advancements in IR, particularly through the **integration of Machine Learning (ML) and Deep Learning (DL)**. [5] While traditional IR systems were rule-based and depended on pre-defined heuristics, modern IR models leverage **data-driven learning approaches** to understand complex relationships between queries and documents, thereby improving retrieval accuracy.

1.1.1 The Role of Evaluation Metrics in Information Retrieval

Evaluating the performance of an Information Retrieval (IR) system is essential to understanding its effectiveness, efficiency, and overall impact on user experience. The core objective of IR is to retrieve the most relevant documents or results in response to a user's query. However, determining whether a system is performing well requires a well-defined framework for assessment. This is where evaluation metrics play a critical role.

Evaluation metrics in IR provide quantitative insights into how well a system ranks, retrieves, and presents information to users. They allow researchers and practitioners to compare different retrieval models, optimize system performance, and make informed decisions regarding algorithmic improvements. Without proper evaluation, an IR system might return results that are seemingly correct but fail to satisfy real-world information needs.

One of the primary contributions of evaluation metrics is their ability to measure **relevance and ranking quality**. [7]Metrics such as **Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain (NDCG), Recall, and Mean Average Precision (MAP)** help determine whether an IR system prioritizes relevant information and presents it in an accessible manner. For instance, in a web search engine, users typically focus on the top few results; hence,

a metric like **MRR** is useful for assessing how quickly a relevant document appears. Meanwhile, **NDCG** accounts for graded relevance, making it particularly valuable for systems where multiple levels of relevance exist. [5]

Moreover, these metrics assist in identifying weaknesses in an IR system. If a model exhibits high precision but low recall, it may retrieve only highly relevant documents but fail to capture the breadth of information a user might need. Conversely, if recall is high but precision is low, the system may return too much irrelevant information, leading to information overload. By analyzing different evaluation metrics, IR practitioners can strike a balance between retrieving sufficient relevant documents and avoiding unnecessary noise.

Another crucial aspect of evaluation metrics is their role in guiding **model development and fine-tuning**. Modern IR models, especially those utilizing deep learning and transformer-based embeddings, require extensive optimization to perform well across various domains. By monitoring metric trends over multiple training iterations, researchers can identify whether a model is overfitting to specific queries, generalizing well across datasets, or failing to capture important semantic nuances.[8]

In addition to assessing retrieval performance, evaluation metrics also play a role in improving **user satisfaction and system usability**. [9] A high-performing IR system should not only retrieve accurate information but also align with user expectations regarding ranking, speed, and diversity of results. Metrics like **user engagement and click-through rate (CTR)** in commercial search engines further enhance traditional IR evaluation by incorporating real-world interactions. [10]

In conclusion, evaluation metrics are indispensable tools in developing and refining IR systems. They enable a systematic and objective comparison of retrieval methods, ensuring that improvements are backed by empirical evidence rather than subjective judgment. As IR continues to evolve with advancements in machine learning and deep learning, these metrics will remain fundamental in shaping the effectiveness and efficiency of next-generation retrieval models.

1.2 Machine Learning and Deep Learning in Information Retrieval

Machine Learning (**ML**) and Deep Learning (**DL**) have become pivotal in advancing **IR** systems, enabling them to go beyond traditional keyword-based retrieval to more **semantic**

and context-aware models. [4] The advent of **ML** and **DL** has significantly impacted the way **IR** systems understand and process **user queries** and **documents**, leading to more efficient and effective retrieval solutions. In this section, we will explore the role of **ML** and **DL** in **IR**, their contributions to the field, and how these techniques are transforming modern **IR** applications. [11]

1.2.1 Machine Learning in Information Retrieval

Machine learning has emerged as a transformative approach to improving traditional **IR** systems. [4] At its core, **ML** enables systems to **learn from data**, identify patterns, and make predictions without being explicitly programmed for every task. In **IR**, this learning process can be applied to both **query processing** and **document retrieval**, improving the system's ability to understand user intent, rank documents, and adapt to different information needs.

The main advantage of using **machine learning** in **IR** is the ability to move from **rule-based** approaches to **data-driven** approaches. [12] For instance, traditional **IR** models rely heavily on predefined rules, such as TF, IDF, and Boolean logic, to rank and retrieve documents. While these models work well for simple queries, they struggle with more **complex, ambiguous, and context-dependent queries**. **ML** techniques, on the other hand, can learn from large datasets and develop a **better understanding** of what constitutes a relevant document for a given query. [13]

In **ML**-based **IR** systems, **supervised learning** and **unsupervised learning** are commonly used techniques. In **supervised learning**, labeled data is used to train the model, where the system learns to associate specific features (e.g., terms or phrases) with relevant or irrelevant documents. **Unsupervised learning**, in contrast, works without labeled data and is used to uncover hidden structures in data, such as **clustering** and **topic modeling**. [14]**Reinforcement learning** is also gaining popularity in **IR** systems, where a model learns by interacting with its environment and receiving feedback about the relevance of the retrieved documents.

Key **ML** algorithms applied in **IR** include **decision trees**, **support vector machines (SVMs)**, and **k-nearest neighbors (KNN)**. These algorithms are used for tasks such as **document classification**, **query classification**, **ranking**, and **relevance feedback**. [15]

1.2.2 Deep Learning in Information Retrieval

DL, a subset of **machine learning**, has brought a significant leap in the capabilities of **IR** systems. Unlike traditional **ML** models that require **manual feature extraction**, **DL** models

are capable of **automatically learning complex features** from raw data, making them particularly suited for tasks in **IR** that involve large, high-dimensional datasets such as text, images, and even multimodal data. [4]

One of the most prominent contributions of **DL** to **IR** is through the use of **neural networks**, particularly **Convolutional Neural Networks (CNNs)** and **Recurrent Neural Networks (RNNs)**, as well as **transformer models** like **BERT (Bidirectional Encoder Representations from Transformers)** and **E5**. These models have revolutionized **semantic search** by learning a **deep understanding** of both query and document representations in a continuous vector space. This allows **IR** systems to go beyond surface-level keyword matching to find semantically relevant results that may not share exact terms but convey similar meanings.

DL in **IR** is particularly effective for tasks such as **semantic document ranking**, **query-document matching**, and **question answering**, where the system needs to **understand the intent behind the query** and match it with documents that may not explicitly contain the search terms. **Transformer-based models**, such as **BERT** and **T5**, use **self-attention mechanisms** to capture long-range dependencies in text, allowing them to better understand the context and meaning of words based on their surrounding words. [16]This ability to capture **contextual meaning** has led to significant improvements in search relevance and user satisfaction.

In addition to transformer-based models, **pre-trained language models** like **BERT** have become essential tools in modern **IR** systems. These models are pre-trained on vast amounts of text data and then fine-tuned on domain-specific tasks, such as document retrieval. The fine-tuning process allows the model to **specialize** in a particular task, improving retrieval performance on domain-specific queries.

Another application of **DL** in **IR** is **reinforcement learning**, where models are trained through feedback mechanisms, learning to improve their search results over time. This approach is particularly useful in **personalized search engines** and **recommendation systems**, where user interactions with the system inform future query ranking decisions.

1.2.3 The Role of ML and DL in Modern IR

The integration of **ML** and **DL** into **IR** has led to a transformation in the field, providing **more accurate, efficient, and user-centric retrieval models**. These methods allow systems to not only retrieve relevant documents but also **understand user preferences** and adapt their results accordingly. The combination of **ML** and **DL techniques** with traditional **IR** methods

creates a **hybrid approach** that leverages the strengths of both worlds. [17]

In particular, **ML** and **DL** enable **IR** systems to perform better in several challenging areas, including **complex queries**, **semantic understanding**, and **multilingual retrieval**. [18] Moreover, these techniques have been instrumental in **adapting to evolving user behavior**, as they allow systems to continuously learn from user interactions and improve their accuracy over time.

In recent years, **end-to-end neural ranking models** have emerged, where the entire retrieval process—from understanding the query to ranking the documents—is handled by deep neural networks. [19] This approach significantly reduces the need for manual feature engineering and allows **IR** systems to focus more on **learning patterns** from data, making them more adaptable to new domains and datasets.

In summary, **ML** and **DL** have greatly enhanced the performance of modern **IR** systems, allowing them to provide more relevant, personalized, and context-aware search results. As these technologies continue to evolve, it is expected that the capabilities of **IR** systems will expand, offering more sophisticated solutions in a wide range of applications.

1.2.4 Integration of ML/DL with Information Retrieval Models

As we proceed in this research, we will investigate the integration of various **ML** and **DL** techniques into existing **IR** models. Specifically, we will focus on **fine-tuning pre-trained models** such as **BGE-Small**, **BGE-Base**, and **E5** with domain-specific datasets to improve retrieval performance. By leveraging the power of **transformer-based models** and fine-tuning strategies, we aim to demonstrate how modern **ML** and **DL** approaches can further enhance the effectiveness of **IR** systems in specialized domains, such as **financial document retrieval**.

1.3 Information Retrieval in Modern Times

In recent years, the landscape of Information Retrieval (**IR**) has evolved significantly, driven by the rapid advancements in **machine learning** (**ML**), **deep learning** (**DL**), and **natural language processing** (**NLP**). Today, **IR** plays a crucial role in numerous real-world applications, ranging from web search engines to personalized recommendation systems and enterprise search solutions. [20]This section will explore how **IR** has adapted to modern challenges, how it has

integrated with other advanced technologies, and how it continues to shape user experiences and drive innovation in the digital age.

1.3.1 Evolution of Information Retrieval Systems

The core principles of **IR** have remained largely unchanged since the early days of search engines, where systems focused primarily on matching keywords in a query with the words in a document. However, over the past few decades, **IR** has undergone a significant transformation due to the integration of more advanced techniques, such as **machine learning** and **deep learning**, that enable systems to understand not just the literal meaning of words but also their **context** and **semantic relationships**.

One of the most notable trends in modern **IR** is the shift from traditional **keyword-based retrieval** methods to **semantic search** approaches. [21] While traditional systems rely on exact keyword matching, **semantic search** systems are designed to understand the meaning behind the user's query and return documents that are contextually relevant, even if they do not share the same keywords. This shift has been facilitated by advancements in **NLP** and the development of models such as **BERT**, **GPT**, and **T5**, which are pre-trained on vast amounts of text data and can capture deep semantic relationships.

The integration of **ML** and **DL** models into **IR** systems has made it possible to move beyond simple rule-based ranking mechanisms to more sophisticated, data-driven approaches. For example, modern **IR** systems now use **neural ranking models** that can learn to rank documents based on relevance, even in the absence of explicit keyword matching. These models can consider a wide range of factors, such as the **query's intent**, **user preferences**, and **document context**, resulting in much more accurate and personalized search results.

1.3.2 IR in the Era of Big Data and Personalization

One of the key factors driving the evolution of modern **IR** is the explosion of **big data**. The amount of digital content being generated worldwide is growing at an exponential rate, leading to an ever-increasing volume of data that needs to be processed, indexed, and retrieved. Traditional **IR** systems, which were primarily designed to handle relatively small, static datasets, are ill-equipped to deal with the scale and complexity of **big data**. [22]

Modern **IR** systems, powered by **ML** and **DL**, are more capable of handling large-scale data and can deliver faster, more relevant results. For example, search engines like **Google**, **Bing**, and **Yahoo** now use sophisticated ranking algorithms that take into account millions of fac-

tors, including **user behavior**, **click patterns**, and **content quality**, in addition to traditional content-based ranking. [23]

Personalization is another major trend in modern **IR**. Personalized **IR** systems use machine learning to analyze users' search history, click behavior, and other preferences to tailor search results to individual users. For instance, **recommendation systems** used by companies like **Netflix**, **Amazon**, and **Spotify** are a form of personalized **IR** that suggests products, movies, music, and other items based on users' past interactions with the platform.

The integration of **user-centric models** into **IR** has made systems smarter and more responsive to users' needs. As **personalized search** continues to grow, future **IR** systems are likely to leverage even more advanced models, such as reinforcement learning, to optimize the user experience.

1.3.3 Challenges and Opportunities in Modern Information Retrieval

Despite the progress made, modern **IR** faces several challenges, particularly in areas such as **bias** in search results, **privacy concerns**, and the handling of **multimodal data**. As **IR** systems increasingly rely on **ML** and **DL** models, there is a growing concern about the transparency and fairness of these systems. Models are often seen as "black boxes," making it difficult to understand how decisions are made, which can lead to biased or unfair search results. Addressing these challenges requires ongoing research into **ethical AI**, fairness in machine learning, and methods for **explainability**. [24]

Another challenge faced by modern **IR** systems is the need to handle **multimodal data**—data that comes in multiple forms, such as text, images, audio, and video. Traditional **IR** systems, which primarily deal with text data, must now be augmented with techniques to process and integrate these diverse data types. **Multimodal retrieval** aims to bridge this gap by developing models that can understand and search across different types of media.

Despite these challenges, the future of **IR** is full of opportunities. Advances in **neural retrieval models**, **transformer architectures**, and **reinforcement learning** provide exciting prospects for even more effective **search systems**. The growing use of **AI** in **IR** has the potential to revolutionize not just how we search for information, but how we interact with technology in everyday life. **IR** will continue to play an integral role in fields such as **e-commerce**, **healthcare**, **finance**, **social media**, and **education**, where the demand for fast, accurate, and personalized information retrieval solutions is ever-growing.

1.3.4 IR and the Future of Search

Looking ahead, the future of **IR** will likely be shaped by the continuing evolution of **AI**, **ML**, and **DL** techniques. As we move toward more interactive and conversational systems, the next generation of **IR** systems will need to support **voice search**, **chatbots**, and **virtual assistants**, offering a more natural, user-friendly way to interact with information. The integration of **AI** with **search engines** and **digital assistants** will enable even more personalized and contextual search experiences.

In conclusion, modern **IR** has evolved into a sophisticated field that is integral to many aspects of our digital lives. By combining **traditional IR techniques** with **machine learning** and **deep learning**, we are witnessing the development of systems that can understand and retrieve information with increasing accuracy, efficiency, and personalization. As **AI** continues to advance, the capabilities of **IR** will expand, enabling even more powerful search technologies that will shape the future of how we access and interact with information.

1.4 Usage of Prefixes in IR Models

In recent years, **prefixes** have emerged as a powerful tool in enhancing the performance of **pretrained models** within **Information Retrieval (IR)** systems. Prefixes typically consist of specific, domain-relevant instructions or cues that are added to the beginning of a query or document during **fine-tuning**. [25] These prefixes provide the model with contextual information that guides its understanding of the specific task or domain. For example, in domain-specific IR tasks, such as financial or legal document retrieval, adding a prefix like "Find the relevant financial news articles related to" before a query can help the model focus on extracting the most pertinent results from the corpus.

Prefix-driven fine-tuning has gained increasing attention because it can significantly improve a model's ability to handle specialized tasks. By guiding the model with domain-specific cues, prefixes enhance the semantic understanding of the text, leading to better contextual relevance in retrieval results. [26] The model learns to associate these prefixes with the specific intent or goal of the query, thus improving its performance in real-world search scenarios. This approach is particularly useful when dealing with **highly specialized domains** where generic pretraining may not provide sufficient accuracy.

1.5 Motivation Behind Prefix-Driven Fine-Tuning

The motivation for using **prefix-driven fine-tuning** in IR tasks stems from the challenge of adapting general-purpose language models to domain-specific problems. While large-scale language models, such as BERT and GPT, are pretrained on vast amounts of general data, they may not capture the nuances and specialized terminology of specific industries like **finance**, **healthcare**, or **law**. Prefix-driven fine-tuning addresses this issue by providing the model with explicit domain-relevant information at the beginning of a query or passage.

In the context of this research, adding a domain-specific prefix allows the model to focus on the particularities of the **FiQA dataset**, which involves **financial questions and answers**. By applying different prefix strategies during fine-tuning, we aim to explore how such approaches can improve the model's ability to retrieve accurate and relevant documents within the financial domain. This experiment helps to confirm whether prefix-driven fine-tuning can effectively enhance the model's adaptability and retrieval performance, especially for complex queries that require domain-specific knowledge.

The remainder of this thesis is structured as follows: **Chapter 2** provides a comprehensive background on the fundamental concepts and models relevant to this research. It delves into the working mechanisms of **sequence-to-sequence (Seq2Seq) models**, the **transformer architecture**, and widely used pretrained models such as **BERT, SBERT, BGE, and E5**. This chapter aims to establish a theoretical foundation for understanding how these models contribute to **information retrieval (IR) tasks**. **Chapter 3** outlines the **research objectives, dataset details, and model selection process**, highlighting the rationale behind choosing specific datasets and fine-tuning approaches. **Chapter 4** presents the **experimental setup, evaluation metrics, and results**, offering a detailed analysis of how different models perform under various fine-tuning and prefix configurations. Finally, **Chapter 5** concludes the thesis by summarizing key findings, discussing limitations, and suggesting potential directions for future research.

2

Transformers Architecture

2.1 Seq2Seq

Seq2Seq (Sequence-to-Sequence) is a neural network architecture designed to map an input sequence to an output sequence. It consists of two main components: an **encoder** and a **decoder**. The encoder processes the sequential input data and transforms it into a fixed-length hidden representation, commonly referred to as the **context vector**. The decoder utilizes this context vector to generate the output sequence. In Figure 2.1, the Seq2Seq architecture process is illustrated with an example.

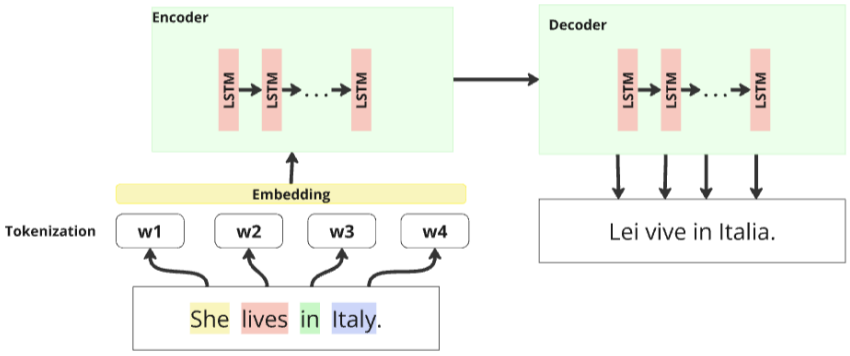


Figure 2.1: Seq2Seq Architecture

The Seq2Seq model was first introduced by **Sutskever et al. (2014)** [27] and **Cho et al. (2014)** [28] with the primary objective of optimizing **machine translation** tasks. Prior to Seq2Seq, traditional models struggled with variable-length input and output sequences. Classical machine learning techniques, such as **Hidden Markov Models (HMMs)** and **Statistical Machine Translation (SMT)**, were ineffective in capturing long-range dependencies. The encoder-decoder structure in Seq2Seq enables the model to process sequences of varying lengths, making it particularly advantageous for natural language processing (NLP) tasks.

The encoder is the initial component of the Seq2Seq architecture, and its specific operations depend on the task. In applications such as **machine translation** and **text summarization**, the encoder typically follows two primary steps:

1. **Tokenization:** Each word in the input sentence is converted into numerical values that mathematically represent them. Tokenization can be performed at the **word level** or **character level**, depending on the requirements of the task.

2. **Hidden State Computation:** The numerical representations are processed through a series of layers that reduce dimensionality while preserving relevant semantic information. The **Recurrent Neural Network (RNN)** or its variants, such as **Gated Recurrent Unit (GRU)** or **Long Short-Term Memory (LSTM)**, are commonly used in this step.

Given an input sequence:

$$X = \{x_1, x_2, \ldots, x_T\} \tag{2.1}$$

where x_t represents each token (word or character) at time step t, the encoder updates its hidden state h_t using the following recurrence relation:

$$h_t = f_{\text{enc}}(x_t, h_{t-1}) \tag{2.2}$$

where:

- h_t is the hidden state at time step t,
- f_{enc} is the recurrent function (RNN, LSTM, or GRU),
- x_t is the input at time step t,
- h_{t-1} is the hidden state from the previous time step.

The final hidden state h_T, often referred to as the **context vector**, encapsulates the encoded information from the input sequence. This context vector serves as the input to the decoder.

The decoder utilizes the context vector to generate the output sequence:

$$Y = \{y_1, y_2, \ldots, y_{T'}\} \tag{2.3}$$

where y_t represents the predicted token at time step t.

The decoder operates sequentially, generating one token at a time. At each time step t, the decoder receives:

- The **previously generated token** (or, during training, the ground truth token in a technique called **teacher forcing**).
- The **hidden state from the previous step**, which carries past information.

The hidden state s_t is updated as follows:

$$s_t = f_{\text{dec}}(y_{t-1}, s_{t-1}) \tag{2.4}$$

where f_{dec} represents the decoder's recurrent function.

At each time step, the decoder generates a probability distribution over the vocabulary using a **softmax layer**:

$$\hat{y}_t = \text{softmax}(Vs_t + b) \tag{2.5}$$

where:

- V and b are learned parameters,
- $\hat{y}_t$ is the probability distribution over the vocabulary.

The token with the **highest probability** is selected as the output at each time step. The process continues iteratively until the model generates a special **end-of-sequence token** <EOS>, signaling the completion of the sequence.

While the Seq2Seq model introduces significant improvements in sequence-to-sequence tasks, it still exhibits several limitations. One of the most critical issues is the **fixed-length bottleneck problem**. Since the encoder stores the entire input sequence information in a single fixed-length context vector, this can lead to information loss, particularly in long sequences.

Another major limitation is the **lack of parallelization**. The model processes tokens sequentially, one at a time, which results in slower decoding and increased computational cost.

Additionally, the Seq2Seq model suffers from the **vanishing gradient problem**. In this phenomenon, gradients diminish exponentially as they propagate backward through time, making it difficult to learn long-range dependencies. As a result, in Seq2Seq models, early tokens in the input sequence may not have a sufficiently strong influence on the generated output, leading to reduced performance in handling long sequences.

Figure 2.2 illustrates that the decoder processes each token separately, requiring all tokens to be stored in memory throughout the entire process. This leads to the issues discussed above.

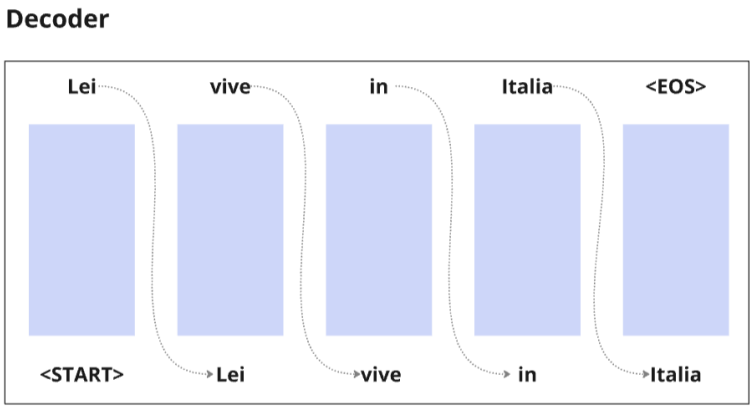


Figure 2.2: Seq2Seq Decoder

Bahdanau et al. (2015) [29] introduced the attention mechanism to address the limitations of traditional seq2seq models. This mechanism allows the model to focus on relevant parts of the input sequence at each decoding step.

The attention mechanism employs dynamic weighting, which allows the model to focus dynamically on different parts of the input sequence while processing the output sequence. Instead of relying solely on a fixed-length context vector to encode the input sequence, the attention mechanism assigns different weights to various parts of the input sequence based on their relevance to the current step of the output sequence. This provides a solution to one of the primary problems of classical seq2seq models.

Moreover, it creates a soft alignment between the input and output sequences by computing a distribution of weights over the input sequence. This represents a substantial advantage, enabling the model to process multiple elements simultaneously. All of these improvements

allow the model to adapt to longer input sequences without significantly increasing computational complexity.

Additionally, because the attention weights represent the model's decision-making process, researchers can visualize these representations to understand which parts of the input sequence are most relevant.

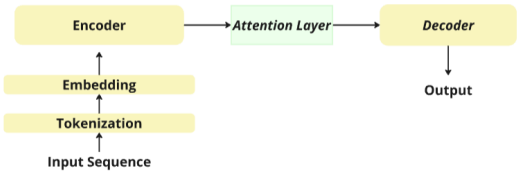


Figure 2.3: Attention Mechanism

2.2 Transformer Architecture

Vaswani, A., et al. (2017) [1] introduced the Transformer architecture with the purpose of improving performance through greater parallelization. To achieve parallelization, a self-attention mechanism is employed instead of relying on RNNs. The Transformer comprises an encoder and a decoder, similar to the seq2seq model. In the original Transformer model, the encoder consists of 6 layers, and the decoder has the same number of layers; however, this number is a design choice and can be adjusted based on the model's requirements. The complete Transformer architecture, including the encoder and decoder, is presented in Figure 2.4.

The encoder begins by encoding the input sequence. After this initial encoding, positional embeddings are added to the token embeddings to inject positional information. Since the Transformer architecture does not use RNNs, there is no inherent recurrence to capture the order of tokens. For this reason, explicit positional information is incorporated into the embeddings. The authors utilize sinus and cosine functions for this purpose, with the sine function generating values for even indices and the cosine function for odd indices. The basic mathematical formulas for the positional encoding are shown in formulas 2.6 and 2.7.

$$PE_{(pos,2i)} = \sin\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right), \tag{2.6}$$

$$PE_{(pos,2i+1)} = \cos\left(\frac{pos}{10000^{\frac{2i}{d_{model}}}}\right), \tag{2.7}$$

where:

- pos is the position of the word in the sequence,
- i is the dimension index of the embedding vector,
- d_{model} is the total dimension of the model,
- $10000^{\frac{2i}{d_{model}}}$ is a scaling factor that ensures different frequencies for different embedding dimensions.

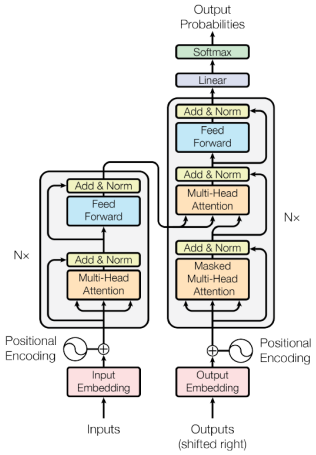


Figure 2.4: Transformers Architecture [1]

The encoder layer consists of multi-headed attention followed by a fully connected network. Each component is succeeded by a normalization layer. Multi-headed attention is essentially a self-attention layer, where self-attention allows the model to weigh the relevance of each word in the input relative to the other words. The representation of this process and its relevance across different layers can be observed in Figure 2.5, which is an example from the research by Vaswani et al. (2017) [1].

In self-attention, the model calculates three vectors for each token: Query (Q), Key (K), and Value (V). These vectors are created by multiplying the token's embedding by three matrices learned during training. The resulting vectors are then fed through a linear layer.

The purpose of self-attention is to determine the relevance of each word with respect to the other words in the sequence. To achieve this, a dot product is computed between the query and key vectors, producing a score matrix that indicates the relationship between tokens. Higher scores correspond to a stronger relationship between the tokens.

Since the dot product can scale the values and adversely affect the gradients, the scores are scaled down by dividing by the square root of the dimension of the query and key vectors, denoted as d_{model}. The scaled score matrix is then passed through a softmax function to obtain the attention weights, essentially probability values between 0 and 1. In the final step of the multi-headed attention, these attention weights are multiplied by the value vectors, resulting in the output vector and the mathematical representation of this process can be observed on Formula 2.8.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_{\text{model}}}}\right)V \tag{2.8}$$

The special aspect of multi-headed attention is that the same input vectors are split into N identical sets, each processed independently through the self-attention mechanism. Although the input vectors are the same, each head uses different learnable weight matrices. Each head produces an output vector, and these output vectors are concatenated into a single vector before being passed through a final linear layer. This method enables each head to learn different aspects of the input.

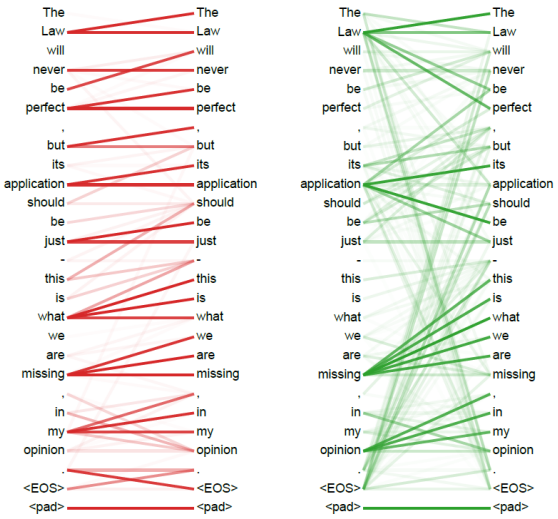


Figure 2.5: Relevance of Words in Two Different Encoder Self-Attention Heads at Different Layers [1]

As previously mentioned, after the multi-headed attention layer and the feed-forward network, the *Add & Norm* step is applied. In this step, the output vector from the multi-headed attention layer is added to the original positional input embedding. This residual connection

helps in stabilizing the learning process and mitigating the vanishing gradient problem. Subsequently, layer normalization is applied to normalize the activations before they are passed to the next component.

The next step in the encoder is the **position-wise feed-forward network (FFN)**, which consists of two fully connected layers with a ReLU activation in between. The primary function of the FFN is to introduce **non-linearity and feature transformation** at each token position independently. This enables the model to learn complex representations beyond the capabilities of linear attention mechanisms. Although the layers in the FFN are linear, the presence of the ReLU activation ensures non-linearity in the transformation.

After processing through the feed-forward network, the *Add & Norm* step is applied again to stabilize the learning process.

The decoder follows a structure similar to the encoder. Its primary purpose is to generate text sequences. It consists of two multi-headed attention layers, a pointwise feed-forward layer, residual connections, and layer normalization after each sub-layer. Each multi-headed attention layer serves a distinct purpose, while the remaining sub-layers are similar to those in the encoder. As in the encoder, the decoder is stacked N times, where $N = 6$ in the original Transformer model.

The first step in the decoder involves embedding the input sequence and adding positional encodings to incorporate positional information. Next, the first multi-headed attention mechanism, known as **Masked Multi-Head Self-Attention**, is applied. In this step, self-attention is performed with a look-ahead mask to prevent information leakage from future tokens. The mask is added before the softmax function is applied. The masking matrix consists of values of 0's and negative infinities, where negative infinity is assigned to future tokens in the sequence. The modified formula is shown in Formula 2.9.

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^T}{\sqrt{d_{\text{model}}}} + \text{mask}\right)V \tag{2.9}$$

Following the masked attention layer, residual connections and layer normalization are applied, as in the encoder. After this, the second multi-headed attention layer is introduced. While the process follows standard self-attention, an important distinction is the input used for each component. Specifically, the query vectors are derived from the decoder's previous self-attention output, while the key and value vectors originate from the encoder's output. This allows the attention mechanism to establish relevance between the decoder's current state and the encoded input sequence. After this step, residual connections and layer normalization are

applied again. The output is then passed through a point-wise feed-forward layer, following the same process as in the encoder.

In the final stage, a linear layer acts as a classifier. Its output dimension corresponds to the vocabulary size. The classifier output is then passed through a softmax layer, producing probability scores for each token. The token with the highest probability is selected as the predicted word.

While Transformers effectively address key limitations of previous sequence-to-sequence models through multi-headed attention, they require substantial computational resources, particularly for large-scale models. Additionally, they demand extensive training data to achieve optimal performance.

2.3 Bidirectional Encoding Representations from Transformers (BERT)

Bidirectional Encoder Representations from Transformers (BERT) is a deep learning model introduced by **Devlin et al. (2018)** at Google [2]. It differs from previous models due to its bidirectional processing. Unlike earlier models that relied on left-to-right or right-to-left context, BERT captures contextual information from both directions simultaneously. This feature makes BERT more powerful for language-understanding tasks and various NLP applications.

BERT is based on the **Transformer** architecture but utilizes only the encoder component. There are two main reasons for this: (1) Decoders are optimized for autoregressive generation, which is not BERT's primary focus, and (2) BERT is designed to understand input sequences rather than generate output sequences. Autoregressive models generate tokens one by one, typically in a left-to-right manner, whereas BERT processes entire sequences bidirectionally. BERT enhances the performance of multiple NLP tasks, including **Text Classification, Named Entity Recognition (NER), Question Answering, and Natural Language Inference**.

BERT's training consists of two steps:

- **Pre-training** on large amounts of unlabeled text to learn contextual embeddings.
- **Fine-tuning** on labeled data for specific NLP tasks.

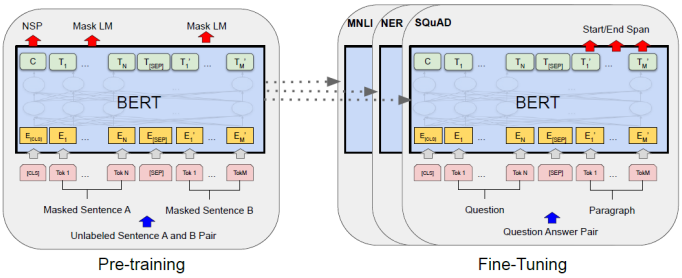


Figure 2.6: Illustration of the BERT pre-training and fine-tuning process. The model undergoes unsupervised pre-training on large text corpora before being fine-tuned on task-specific datasets for downstream applications [2]

During training, token sequences are fed into the Transformer encoder, where they are first embedded into vectors and then processed through a neural network. BERT's input representation is composed of three key embedding types: **Token Embeddings, Segment Embeddings, and Positional Embeddings**. The BERT's input representation is shown in Figure 2.7. **Token Embeddings** represent the individual words or subwords in the input sequence. **Segment Embeddings** differentiate between sentences when multiple sentences are provided as input; for example, in Next Sentence Prediction (NSP), Sentence A and Sentence B are assigned distinct segment IDs. **Positional Embeddings** encode the position of each token in the sequence, ensuring that the Transformer model can capture the order of words, even though it does not process input sequentially. These three embeddings are summed element-wise to form the final input representation, which is then passed through the Transformer encoder for processing. Unlike previous models that used a directional approach by predicting the next word in a sequence, BERT overcomes this limitation by introducing two training strategies: **Masked Language Model (MLM)** and **Next Sentence Prediction (NSP)**.

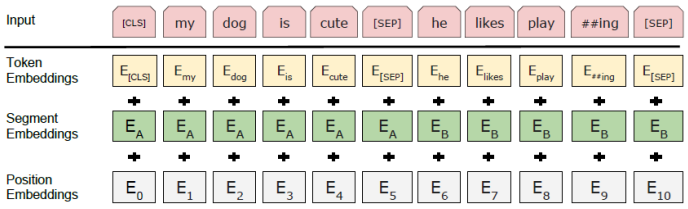


Figure 2.7: Illustration of BERT's input representation, including token embeddings, positional embeddings, and segment embeddings. These components enable the model to capture contextual meaning, word order, and sentence distinctions for effective language understanding. [2]

In **MLM**, 15% of the tokens in the input sequence are randomly masked. The model is then trained to predict the original values of these masked words based on the surrounding context. When masking words, BERT replaces them with a special token, **[MASK]**. A classification layer on top of BERT's encoder structure makes these predictions. The model evaluates its accuracy by comparing predictions to the original masked words. The output vectors are projected into the vocabulary space using an **embedding matrix**, and a **SoftMax function** generates a probability distribution over the vocabulary. The loss function penalizes incorrect predictions but focuses only on the masked tokens. Importantly, BERT does not predict unmasked words, which allows it to excel in understanding word meanings and contextual relationships. However, this bidirectional approach results in slower convergence than directional models since BERT focuses on masked token prediction. In **Next Sentence Prediction (NSP)**, the goal is to determine whether two sentences are logically connected. Specifically, the model predicts whether the second sentence follows the first in a coherent manner. This step utilizes special tokens: **[CLS]** and **[SEP]**. The **[CLS]** token is inserted at the beginning of the first sentence, while a **[SEP]** token is added at the end of each sentence. Additionally, sentence embeddings and positional embeddings are incorporated to preserve contextual relationships.

In the Transformer architecture, particularly in models like BERT, the input consists of two sentences, and the model is trained to predict whether the second sentence logically follows the first in a given pair. The model processes these sentences and generates embeddings for each token. The output corresponding to the special `[CLS]` token, which represents the entire sentence pair, is then passed through a linear transformation layer. This transformation reduces the output's dimensionality to a 2-dimensional vector, where each dimension represents the likelihood of one of two possible outcomes: the first value indicates the probability that the second sentence follows the first (i.e., the pair is consecutive), while the second value represents the probability that the second sentence does not follow the first (i.e., the pair is randomly paired). This 2-dimensional vector is passed through a Softmax function, which converts these raw values into probabilities, ultimately allowing the model to classify the sentence pair as either consecutive or random. During training, half of the sentence pairs are actual consecutive sentences, while the remaining pairs are randomly paired, thus enabling the model to learn the relationship between consecutive and non-consecutive sentence pairs.

BERT is trained using both **MLM** and **NSP**, optimizing a combined loss function to enhance **contextual understanding** and **sentence relationship modeling**.

2.4 Sentence Bidirectional Encoding Representations from Transformers - SBERT

S-BERT is a modified version of BERT designed specifically for sentence similarity and embedding tasks. **Sentence-BERT (SBERT)** was introduced in **2019** by **Nils Reimers and Iryna Gurevych** from the **UKP Lab at the Technical University of Darmstadt, Germany** [3]. Unlike BERT, which generates contextualized token embeddings, SBERT produces fixed-size sentence embeddings that can be efficiently compared using cosine similarity. SBERT was introduced to address the **quadratic complexity** of BERT, which requires pairwise comparisons, making it slow for large datasets. SBERT fine-tunes BERT using **Siamese** and **Triplet Networks**, generating meaningful sentence embeddings without requiring pairwise encoding.

The main idea of a **Siamese Network** is to encode each sentence separately and then compare their embeddings. The process begins by passing input sequences through BERT, where two sentences are encoded separately using a shared BERT model. This step ensures that both sentences obtain contextualized embeddings without direct interaction. The initial progress can be observed from the Figure 2.8.

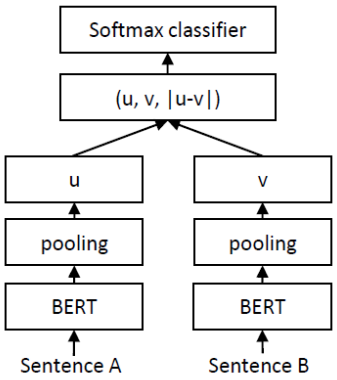


Figure 2.8: SBERT architecture for sentence pair classification. Sentences are encoded with BERT, pooled into fixed-size embeddings, and combined before passing through a softmax classifier. [3]

A crucial step in the Siamese network is the **Pooling Layer**. Since SBERT produces fixed-size sentence embeddings to be efficiently compared using cosine similarity, the pooling layer converts token embeddings into a fixed-size sentence representation. Because BERT outputs embeddings for each token, reducing them into a single vector is necessary.

In SBERT, there are three main pooling techniques used to obtain a fixed-size sentence embedding:

- **CLS Pooling**: Uses the **[CLS]** token's embedding as the sentence representation. The **[CLS]** token is added at the start of every sentence during input tokenization. BERT's final transformer layer processes it, meaning its embedding theoretically represents the entire sentence.

  Advantages: Fast and efficient, as it extracts only a single token. It works well for classification tasks since BERT was trained with **[CLS]** for classification.

  Disadvantages: Not optimal for semantic similarity tasks because the **[CLS]** embedding often focuses on task-specific information rather than full semantic meaning. Additionally, in deeper BERT layers, **[CLS]** may lose critical information from other tokens.

- **Mean Pooling**: Computes the average of all token embeddings, excluding **[CLS]** and **[SEP]**. Since it considers all token representations, it is more effective for sentence similarity tasks. It is commonly used when high-quality sentence embeddings are required for downstream tasks such as semantic search and information retrieval.

- **Max Pooling**: Selects the maximum value for each embedding dimension across all tokens. This means each dimension of the final sentence embedding is determined by the most dominant feature from any word in the sentence. While it captures the most important information, it can lose contextual details. It is useful when key words matter most.

After the pooling layer, **cosine similarity** is used to compare sentence embeddings. Cosine similarity is computed using the following formula:

$$\cos(\mathrm{u}, \mathrm{v}) = \frac{u \cdot v}{\|u\|\|v\|} \tag{2.10}$$

where u and v are the sentence embeddings. The cosine similarity score ranges from -1 to 1, indicating how semantically similar the two sentences are.

SBERT utilizes different loss functions depending on the task. These functions define how the model learns embeddings for tasks such as semantic similarity, retrieval, and classification. The most common loss functions include:

- **Contrastive Loss**: Used in Siamese networks for similarity learning. It calculates the **Euclidean distance** between two sentence embeddings. If the sentences are similar, the loss function minimizes the distance; if they are dissimilar, it maximizes the margin, pushing them apart. It is commonly applied in sentence similarity tasks, text matching, and paraphrase detection.

- **Triplet Loss**: Uses three embeddings—**Anchor** (a reference sentence), **Positive** (a semantically similar sentence), and **Negative** (an unrelated sentence). The goal is to make the Anchor and Positive embeddings closer together while pushing the Anchor and Negative embeddings farther apart. It is widely used in sentence similarity ranking, semantic search, and text retrieval. The Triplet Loss function is defined as:

$$L_{triplet} = \max\left(\|\mathbf{a} - \mathbf{p}\|_2^2 - \|\mathbf{a} - \mathbf{n}\|_2^2 + \alpha, 0\right)$$

Where: - $\mathbf{a}$ is the anchor (reference sentence), - $\mathbf{p}$ is the positive sample (similar sentence), - $\mathbf{n}$ is the negative sample (dissimilar sentence), - α is a margin hyperparameter that ensures the negative sample is sufficiently far from the anchor.

- **Cosine Similarity Loss**: Maximizes cosine similarity for positive sentence pairs and minimizes it for negative pairs. Since SBERT primarily compares embeddings using cosine similarity rather than Euclidean distance, this loss function is effective for ranking similarity scores. The Cosine Similarity Loss function is defined as:

$$L_{cosine} = 1 - \cos(\mathbf{u}, \mathbf{v}) = 1 - \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \, \|\mathbf{v}\|}$$

Where: - $\mathbf{u}$ and $\mathbf{v}$ are the two sentence embeddings, - $\cos(\mathbf{u}, \mathbf{v})$ is the cosine similarity between the two vectors.

- **Multiple Negative Ranking Loss (MNR)**: Commonly used in information retrieval. Unlike Triplet Loss, which uses only one negative sample per triplet, MNR compares each positive example against all other negatives in the batch. This approach speeds up training and improves effectiveness by dynamically selecting negative samples. Additionally, it uses a **SoftMax loss** to ensure the correct sentence is ranked higher than incorrect ones. The MNR Loss function is defined as:

$$L_{MNR} = -\log\left(\frac{\exp(\mathbf{q} \cdot \mathbf{p}_+)}{\sum_{i=1}^{N} \exp(\mathbf{q} \cdot \mathbf{p}_i)}\right)$$

Where: - $\mathbf{q}$ is the query embedding, - $\mathbf{p}_+$ is the positive (relevant) passage embedding, - $\mathbf{p}_i$ represents a set of negative (irrelevant) passage embeddings, - N is the number of negative samples.

2.5 E5 Model

E5 stands for **Embedding-based Efficiently-trained Encoder**, a retrieval-focused embedding model introduced by researchers at Microsoft Corporation. The E5 model was primarily developed for dense retrieval tasks, such as search, question answering, and retrieval-augmented generation. It represents a significant advance in embedding techniques, with a key focus on optimizing retrieval performance. The original E5 model was proposed in [30], and since its initial development, Microsoft has continued to enhance it, resulting in more efficient and faster versions, including E5 v2, which provides even higher accuracy and retrieval efficiency.

The E5 model is built on the principles of **weakly-supervised contrastive learning**, which allows it to effectively distinguish between similar and dissimilar text pairs. The training process involves a large-scale dataset called **CCPairs**, which contains high-quality pairs of text. Traditional approaches to contrastive learning rely on sparse labels or synthetic data, which can result in lower-quality representations. In contrast, the E5 model utilizes a curated, web-scale dataset, ensuring that the embeddings are trained on high-quality data that reflect real-world text relationships. To further improve the quality of the dataset, the authors employed a **consistency-based filter**, removing low-quality pairs to ensure the final training data was highly relevant and semantically rich. This approach enables the E5 model to learn more meaningful and context-aware representations of text.

One of the most important innovations in the E5 model is its use of **supervised fine-tuning**, which allows the model to further enhance its performance on specific tasks. After the contrastive pre-training phase, the E5 model undergoes fine-tuning using labeled datasets. For this purpose, three key datasets were selected: **NLI (Natural Language Inference)**, **MS-MARCO (Microsoft Machine Reading COmprehension) passage ranking dataset**, and **NQ (Natural Questions)**. The NLI dataset helps train the model to understand the relationship between premise and hypothesis, while MS-MARCO and NQ help the model improve its ability to rank passages for search and retrieval tasks. Fine-tuning with these datasets ensures that the model can generalize effectively to a variety of tasks, particularly those that require high semantic precision, such as question answering and information retrieval.

The training process for E5 also introduces **text instruction-based training**, where task-specific instructions are used to guide the model's understanding of different textual contexts. For example, queries are prefixed with `query:`, and corpus data is prefixed with `passage:`. This approach is particularly advantageous because it allows the model to better understand the contextual meaning of text, as opposed to treating all text uniformly. Traditional embeddings,

such as those from BERT and SBERT, do not differentiate between different types of text inputs, which limits their performance in tasks like retrieval. In contrast, the E5 model, with its instruction-based approach, is able to better match queries with relevant passages, making it highly effective for search and retrieval tasks.

An additional key feature of E5 is its multilingual capability. The team at Microsoft recognized the growing need for cross-lingual models, particularly in multilingual search and retrieval tasks. The E5 model was therefore trained to handle multiple languages, enabling it to provide more accurate retrieval results across different linguistic contexts. This multilingual focus makes E5 particularly suited for global applications, where users may input queries in different languages and expect relevant information from a diverse set of sources. The E5 model's ability to seamlessly handle multiple languages sets it apart from other models, which may struggle to provide accurate retrieval across different languages [31].

Since its initial release, Microsoft has continued to refine and enhance the E5 model. The latest version, **E5 v2**, introduces optimizations that result in faster inference and improved accuracy. E5 v2 also incorporates advancements in efficient training and scalability, enabling it to handle even larger datasets and deliver more responsive retrieval results. These updates make E5 v2 a competitive solution for dense retrieval tasks, particularly in domains like search engines, knowledge retrieval systems, and question-answering applications.

Moreover, the E5 model is available in multiple versions with different parameter sizes, including small, base, large, and XL, providing flexibility in adapting to various computational resources and task requirements. These models cater to different performance needs, from faster inference at the cost of some accuracy to larger models that offer higher accuracy but require more computational power. For further details and access to pretrained models, the E5 model is available on the Hugging Face platform [32].

In summary, the E5 model represents a significant advance in dense retrieval and semantic understanding. Through its combination of weakly supervised contrastive learning, supervised fine-tuning, instruction-based training, multilingual support, and availability in multiple versions, E5 offers superior performance on retrieval tasks. Its continuous improvement, with versions such as E5 v2, and its availability in various sizes, highlights its ongoing potential to drive advancements in efficient and accurate information retrieval across various domains.

2.6 BGE Model

The **BGE Model (Bilingual General Embeddings)** is a **text embedding model** designed for a range of natural language processing (NLP) tasks, including **semantic search, text similarity, and retrieval-based applications**. It was introduced by **BAAI (Beijing Academy of Artificial Intelligence)** [33] to improve the performance of multilingual NLP applications, with a specific focus on **English** and **Chinese**. The BGE model is trained on a large-scale dataset that includes both **monolingual** and **bilingual** data, making it one of the most effective models for handling bilingual tasks, particularly in the context of cross-lingual semantic understanding.

Traditional embedding models, such as BERT and word2vec, often face challenges in handling cross-lingual tasks, especially when working with languages like Chinese. The existing models were not optimized for **English-Chinese bilingual tasks**, and the lack of a dedicated, well-optimized model for Chinese further compounded the issue. The BGE model was explicitly designed to address these challenges by providing superior accuracy, better semantic understanding, and more efficient retrieval capabilities. As a result, BGE outperforms models like SBERT in bilingual scenarios and is highly optimized for retrieval-based applications, making it an ideal choice for multilingual tasks.

The architecture of BGE is transformer-based, similar to BERT and SBERT, but with specialized modifications tailored to bilingual tasks. Like E5, BGE leverages a **contrastive learning objective**, helping the model to distinguish between semantically similar and dissimilar text pairs, which is crucial for tasks such as semantic search.

In the training process, the BGE authors selected a large, high-quality dataset that underwent several stages:

1. **Pretraining with plain texts**,
2. **Weakly-supervised contrastive learning** with an unlabeled dataset,
3. **Multi-task learning** with a labeled dataset.

The training pipeline also incorporated semantic filtering to ensure that text pairs used for training were semantically related, improving the overall training performance.

One of the key innovations of the BGE model is the incorporation of **RetroMAE (Retrospective Masked Autoencoders)** in its pretraining strategy. RetroMAE is a self-supervised learning method designed to enhance text embeddings using a combination of **masked autoencoding** and a **retrospective loss function**. This advanced method improves upon the

traditional **Masked Language Model (MLM)** used in models like BERT, resulting in more accurate and robust embeddings.

Another distinguishing feature of BGE is its use of **instruction-based fine-tuning**. During this phase, the model is trained not only on raw data but also on explicit instructions that guide the model to generate embeddings specific to various tasks. This approach ensures that the model is fine-tuned to better align with tasks like semantic search. Unlike other models that rely solely on sentence pair instructions, BGE is fine-tuned with task-specific instructions, such as:

> "Search relevant passages for the query:"

In addition to using in-batch negatives, BGE introduces a strategy where a **hard negative** is sampled for each training pair. This means that, during fine-tuning, the model learns to distinguish between a query, a positive passage, and a challenging negative passage. This instruction-based fine-tuning allows BGE to achieve superior performance on tasks such as semantic search and retrieval, making it one of the most powerful models in the space.

Furthermore, the BGE model is available in multiple versions, including small, base, and large models, each with different parameter sizes. This flexibility allows users to choose a model that best suits their specific needs, depending on factors like computational resources and performance requirements. Additionally, BGE is optimized for multilingual use cases, supporting tasks involving multiple languages beyond just English and Chinese. This makes BGE a versatile solution for global applications where users may interact with queries in different languages.

For additional details, including access to pretrained models, the BGE model can be found on the Hugging Face platform [34].

2.7 Evaluation Metrics

Evaluation metrics are essential for measuring the effectiveness of information retrieval systems. They provide quantitative insights into how well a system retrieves relevant documents and enable comparisons between different models or configurations. These metrics play a crucial role in guiding improvements, ensuring that retrieval systems achieve a balance between accuracy, relevance, and ranking quality.

Evaluation metrics can be categorized into two groups: **order-unaware metrics** and **order-aware metrics**. [35]

- **Order-unaware metrics** focus only on whether relevant documents are retrieved but do not consider their ranking. Examples include **Precision, Recall,** and **F1-Score**. These metrics are particularly useful when the presence of relevant documents is more important than their position in the ranked list.

- **Order-aware metrics** take into account the ranking of documents, giving higher importance to relevant documents appearing earlier in the list. Examples include **Mean Reciprocal Rank (MRR), Normalized Discounted Cumulative Gain (NDCG),** and **Mean Average Precision (MAP)**. Order-aware metrics better reflect real-world user experiences, as users are more likely to engage with top-ranked results.

Order-aware metrics are often preferred in real-world applications, as they provide a more accurate assessment of ranking quality. They help evaluate how effectively a system ranks relevant documents, making them particularly useful for search engines, question-answering systems, and recommendation engines.

2.7.1 Precision

Precision measures the accuracy of retrieved documents by calculating the proportion of relevant documents among the retrieved set:

$$\text{Precision} = \frac{|\text{Relevant Documents} \cap \text{Retrieved Documents}|}{|\text{Retrieved Documents}|} \tag{2.11}$$

A higher precision score indicates fewer false positives, making this metric essential when users expect highly relevant results at the top.

2.7.2 Recall

Recall quantifies how well a system retrieves all relevant documents:

$$\text{Recall} = \frac{|\text{Relevant Documents} \cap \text{Retrieved Documents}|}{|\text{Relevant Documents}|} \tag{2.12}$$

High recall ensures that fewer relevant documents are missed, which is crucial in applications requiring exhaustive information retrieval.

2.7.3 F1-Score

The F1-Score balances precision and recall using their harmonic mean:

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{2.13}$$

This metric is useful when both false positives and false negatives need to be minimized equally.

2.7.4 Mean Reciprocal Rank (MRR)

MRR evaluates the ranking position of the first relevant document across multiple queries:

$$\text{MRR} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \frac{1}{\text{rank}_i} \tag{2.14}$$

where is the position of the first relevant document for query . MRR is highly useful in question-answering applications, where retrieving the most relevant answer at the top is crucial.

2.7.5 Normalized Discounted Cumulative Gain (NDCG)

NDCG accounts for graded relevance and ranking order:

$$\text{DCG} = \sum_{i=1}^{p} \frac{\text{rel}_i}{\log_2(i+1)} \tag{2.15}$$

$$\text{NDCG} = \frac{\text{DCG}}{\text{IDCG}} \tag{2.16}$$

where is the relevance score at position , and is the ideal DCG for perfect ranking. NDCG is particularly useful in ranking tasks where relevance varies in degree.

2.7.6 Mean Average Precision (MAP)

MAP computes the average precision across multiple queries:

$$\text{MAP} = \frac{1}{|Q|} \sum_{i=1}^{|Q|} \text{AP}_i \tag{2.17}$$

where is the average precision for query . MAP is beneficial in evaluating retrieval systems where multiple relevant documents exist for each query.

Together, these metrics serve as fundamental tools for evaluating and refining information retrieval systems. Order-aware metrics provide deeper insights into ranking quality, making them particularly effective for applications where result order impacts user experience. A comprehensive offline evaluation using these metrics ensures that retrieval models are optimized for both accuracy and ranking quality.

3

Research Objective

In recent years, the rapid expansion of data in high-impact societal domains such as healthcare, education, and finance has led to an increased need for effective and efficient semantic search systems. Traditional search methods, based on keyword matching or basic query understanding, often fail to provide relevant results when the context of the query is domain-specific. As a result, semantic search, which focuses on understanding the meaning behind queries and documents, has become a critical area of research.

Pretrained models, particularly those built on transformer architectures, have demonstrated remarkable success in various natural language processing (NLP) tasks. However, these models are typically trained on general datasets that may not effectively capture the nuances of specific domains. Fine-tuning these models on specialized datasets can enhance their performance in domain-specific applications, but challenges persist in adapting these models for semantic search, particularly in high-impact societal domains where accuracy, relevance, and ethical considerations are paramount.

One promising approach to improve domain-specific semantic search is contrastive learning, a technique that learns representations by maximizing the similarity between positive pairs and minimizing the similarity between negative pairs. However, integrating contrastive learning with pretrained models and leveraging domain-specific data remains an open challenge. Additionally, recent research has explored the use of prefix-based methods to guide model learning by conditioning the representations with domain-specific prefixes, potentially improving semantic search performance.

This research addresses the gap in understanding how prefix-driven contrastive learning can enhance semantic search in high-impact societal domains. It explores the potential benefits of fine-tuning pretrained models using prefix-based techniques on domain-specific datasets and evaluates their effectiveness in improving search relevance, accuracy, and domain adaptation.

3.1 Dataset

The selection of an appropriate dataset is a critical aspect of this research. A clean and well-structured dataset enhances the reliability of experimental results and makes performance improvements more discernible. This is particularly important in the context of large language models (LLMs), where the presence of noisy data can adversely impact experiment outcomes and model performance.

During the dataset selection process, three datasets were chosen for their suitability for this research: **FiQA** [36], **NF Corpus** [37], and the **CODEC** [38] dataset. The primary reason for selecting these datasets is their consistent structure, which includes documents or passages (corpus), queries, and relevance judgments (qrels). Additionally, each of these datasets represents different domains, providing a significant advantage for experimenting with domain-specific prefixes across a diverse set of data.

Furthermore, having distinct splits for training, validation, and testing is an important factor in conducting thorough experiments. While both the FiQA and NF Corpus datasets provide these three splits, the CODEC dataset lacks a predefined split. As a result, we opted to perform the dataset split ourselves using the train-test split operation provided by `Scikit-learn`.

3.1.1 FiQA Dataset

The **FiQA (Financial Question Answering)** [36] dataset is a benchmark dataset specifically designed for evaluating **information retrieval, question answering, and sentiment analysis** within the **financial domain**. It serves as a valuable resource for developing and assessing models in **domain-specific semantic search tasks**, where understanding financial terminology and context is crucial. [39]

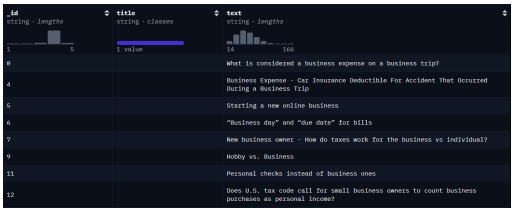
The dataset consists of **financial questions, user-generated content, financial news, and expert-annotated answers** relevant to various queries. The questions are primarily sourced from **online financial forums**, while the answers include both **community responses and expert insights**, making the dataset particularly useful for studying real-world **financial discourse and knowledge retrieval**. Additionally, the dataset includes **QRELs (query relevance**

judgments), which are essential for evaluating retrieval models by determining the relevance of retrieved documents to specific queries.

To ensure **robust model evaluation and prevent overfitting**, the dataset is divided into three distinct subsets:

- **Training set:** Used for fine-tuning models on financial question-answering tasks.
- **Validation (dev) set:** Helps in optimizing hyperparameters and evaluating intermediate performance.
- **Test set:** Serves as the final benchmark for assessing model effectiveness on unseen financial queries.

(a) FiQA Dataset Corpus Example



(b) FiQA Dataset Queries Example

(c) FiQA Dataset Qrels Example

Figure 3.1: Examples from the FiQA Dataset. (a) The corpus consists of financial documents that serve as the retrieval database. (b) The queries represent financial questions or statements used for search evaluation. (c) The Qrels (query relevance judgments) provide mappings between queries and relevant corpus documents, assigning relevance scores for evaluation.

Figure 3.1 illustrates different components of the FiQA dataset, which is used for financial question-answering and information retrieval tasks. The figure 3.1a displays an example from the FiQA corpus, which consists of financial news articles, reports, and discussions that form the collection of documents. The figure 3.1b presents sample queries that represent user search intents, such as financial questions and statements. The figure 3.1c provides an example of Qrels (query relevance judgments), which establish relevance relationships between queries and the corresponding documents in the corpus. These Qrels contain graded relevance scores,

indicating the degree of relevance between a query and a retrieved document. Together, these components allow one to evaluate retrieval models in a structured and reproducible manner.

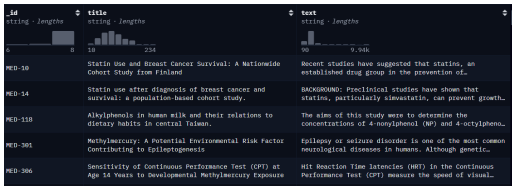
A key advantage of using FiQA [36] is its **inclusion in the Massive Text Embedding Benchmark (MTEB)** [40], a large-scale benchmark designed for evaluating text embedding models across multiple tasks, including retrieval, classification, and reranking. As part of MTEB, FiQA allows for **direct comparison with other pretrained embedding models**, enabling researchers to assess the performance of **domain-specific models against general-purpose ones**. This inclusion provides a standardized evaluation framework, helping to **identify strengths and weaknesses of different approaches** in financial text retrieval.

FiQA is widely used in **financial NLP research** because it reflects the complexity and ambiguity of financial language, requiring models to handle **domain-specific jargon, sentiment variations, and contextual dependencies**. By leveraging this dataset, researchers can enhance the accuracy and relevance of financial information retrieval systems, contributing to more effective **semantic search, automated financial advisory systems, and risk analysis tools**.

3.1.2 NF Corpus Dataset

The **NF Corpus (Neural Feedback Corpus)** [37] is a specialized dataset designed to advance research in **neural information retrieval (IR)**, with a particular focus on **scientific and medical literature**. Unlike traditional keyword-based retrieval tasks, which primarily rely on exact matches between query terms and document content, the NF Corpus facilitates more sophisticated retrieval methods that leverage **neural models** and **feedback signals**. This dataset provides **multiple relevance scores**, which are generated based on **expert annotations** and **neural feedback signals**, allowing for **more nuanced and context-sensitive ranking evaluations**. These relevance scores offer a richer evaluation metric, enabling models to better understand and account for the subtleties inherent in **scientific discourse** and **medical terminology**.

One of the primary challenges in using the NF Corpus is its focus on **domain-specific terminology**, which often includes highly specialized vocabulary and jargon not commonly found in other datasets. This characteristic introduces significant complexity in retrieval tasks, as models must not only understand the syntactic structure of scientific text but also grasp the nuanced **semantic relationships** between concepts. This makes the NF Corpus an ideal benchmark for evaluating models designed for **high-precision, domain-specific information retrieval**, where **traditional IR techniques** may fall short. Figure 3.2 provides example snippets from the NF Corpus dataset. It includes (a) the document corpus, which consists of scientific and



_id	title	text
MED-10	Statin Use and Breast Cancer Survival: A Nationwide Cohort Study from Finland	Recent studies have suggested that statins, an established drug group in the prevention of…
MED-14	Statin use after diagnosis of breast cancer and survival: a population-based cohort study.	BACKGROUND: Preclinical studies have shown that statins, particularly simvastatin, can prevent growth…
MED-110	Alkylphenols in human milk and their relations to dietary habits in central Taiwan.	The aims of this study were to determine the concentrations of 4-nonylphenol (NP) and 4-octylpheno…
MED-301	Methylmercury: A Potential Environmental Risk Factor Contributing to Epileptogenesis	Epilepsy or seizure disorder is one of the most common neurological diseases in humans. Although genetic…
MED-306	Sensitivity of Continuous Performance Test (CPT) at Age 14 Years to Developmental Methylmercury Exposure	Hit Reaction Time latencies (HRT) in the Continuous Performance Test (CPT) measure the speed of visual…

(a) NF Corpus Dataset Corpus Example

_id	title	text
PLAIN-3		Breast Cancer Cells Feed on Cholesterol
PLAIN-4		Using Diet to Treat Asthma and Eczema
PLAIN-5		Treating Asthma With Plants vs. Pills
PLAIN-6		How Fruits and Vegetables Can Treat Asthma
PLAIN-7		How Fruits and Vegetables Can Prevent Asthma
PLAIN-8		Our Tax Dollars Subsidize Unhealthy Foods
PLAIN-9		Reducing Arsenic in Chicken and Rice
PLAIN-10		How Contaminated Are Our Children?

(b) NF Corpus Dataset Queries Example

query-id	corpus-id	score
PLAIN-3	MED-2436	1
PLAIN-3	MED-2437	1
PLAIN-3	MED-2438	1
PLAIN-3	MED-2439	1
PLAIN-3	MED-2440	1
PLAIN-3	MED-2427	1

(c) NF Corpus Dataset Qrels Example

Figure 3.2: Examples from the NF Corpus Dataset. (a) The corpus consists of scientific and medical documents that serve as the retrieval database. (b) The queries represent expert-annotated search queries focusing on biomedical and scientific literature. (c) The Qrels (query relevance judgments) provide expert-annotated relevance scores between queries and corpus documents, allowing for nuanced ranking evaluations.

medical texts, (b) sample queries used for retrieval tasks, and (c) query relevance judgments (Qrels), which indicate the relevance of specific documents to given queries based on expert annotations. These components help evaluate the effectiveness of information retrieval models in domain-specific search tasks. In addition to its application in evaluating neural retrieval systems, the NF Corpus [37] is widely used for **performance benchmarking** within the research community. Notably, it is included in the **MTEB (Multi-Task Evaluation Benchmark) leaderboard** [40], which allows for direct performance comparisons across a range of base models. This inclusion is particularly valuable as it provides researchers with an **objective, standardized measure** of how well their models perform on **real-world, high-impact tasks** in scientific and medical domains. By evaluating base models on this dataset, we can make more informed comparisons and gain deeper insights into the **effectiveness of various techniques**, such as **prefix-driven contrastive learning**, in addressing the challenges posed by complex, domain-specific queries.

Furthermore, the NF Corpus provides a rich resource for conducting experiments that target the improvement of information retrieval in **high-stakes societal domains**, such as **healthcare** and **scientific research**. With its detailed annotations and extensive coverage of **scientific and medical literature**, it offers a unique opportunity to push the boundaries of retrieval techniques in these critical areas.[41]

3.1.3 CODEC Dataset

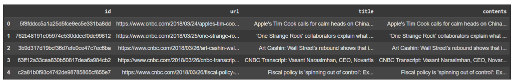
The **CODEC (Complex Document and Entity-centric Corpus)** [38] dataset is designed to advance the evaluation of **information retrieval (IR)** in complex, high-impact societal domains, particularly in **economics**, **history**, and **politics**. This dataset provides a comprehensive collection of **documents** and **queries**, with relevance judgments carefully crafted to assess a model's ability to process and retrieve information from multi-faceted and **contextually rich** content. Unlike traditional IR benchmarks, which primarily rely on keyword matching or surface-level content similarity, CODEC challenges models to go beyond simplistic retrieval methods and engage in deeper **semantic understanding** and **complex reasoning**. [42]

One of the defining features of CODEC is its focus on **complex and multi-dimensional queries**, which often require models to navigate intricate relationships between **entities**, **concepts**, and **contexts** within documents. The dataset is designed to test the ability of models to handle **entity-centric retrieval**, where queries are not only concerned with individual terms but also with **the context** in which these terms are used. For example, a query may require a model to retrieve relevant documents that reference a particular historical figure or political event, but also understand the broader context, such as **historical significance**, **political relationships**, or **economic implications**.

The dataset provides relevance judgments based on **expert annotations**, allowing for a more sophisticated evaluation of a model's ability to retrieve information that is both **contextually relevant** and **semantically accurate**. These expert annotations are critical for assessing how well models can engage with complex and often **ambiguous queries**, a challenge that arises frequently in real-world retrieval tasks in **societal and policy-related domains**.

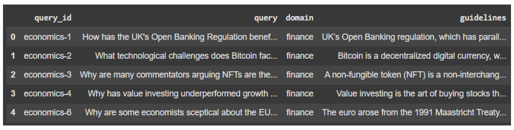
Figure 3.3 provides example snippets from the CODEC dataset. It includes (a) the document corpus, consisting of complex, multi-faceted information from fields like economics and politics, (b) example queries used for evaluation, and (c) query relevance judgments (Qrels), which map queries to relevant documents with assigned relevance scores. These components allow for the evaluation of information retrieval models in handling complex and entity-aware retrieval tasks.

Given its focus on high-stakes topics such as economics, history, and politics, CODEC serves as an important tool for **evaluating the performance** of retrieval models in **real-world scenarios**, where **domain expertise** and **contextual understanding** are paramount. The dataset offers a unique opportunity to benchmark models in scenarios that require not only effective **keyword matching** but also **deep semantic search capabilities**, enabling more accurate re-



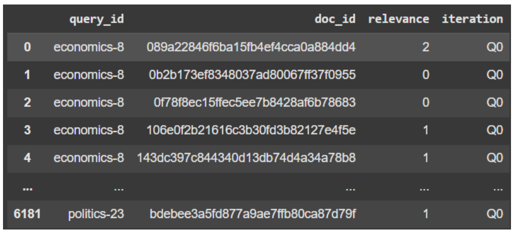
	id	url	title	contents
0	5f8fddcc5a1a25d5fce9ec5e331ba8dd	https://www.cnbc.com/2018/03/24/apples-tim-coo...	Apple's Tim Cook calls for calm heads on China...	Apple's Tim Cook calls for calm heads on China...
1	762b48191e05974e530ddeef0de99812	https://www.cnbc.com/2018/03/25/one-strange-ro...	'One Strange Rock' collaborators explain what ...	'One Strange Rock' collaborators explain what ...
2	3b9d317d19bcf36d7efe0ce47c7ec6ba	https://www.cnbc.com/2018/03/26/art-cashin-wal...	Art Cashin: Wall Street's rebound shows that i...	Art Cashin: Wall Street's rebound shows that i...
3	63ff12a33cea830b50817dea6a984cb2	https://www.cnbc.com/2018/03/26/cnbc-transcrip...	CNBC Transcript: Vasant Narasimhan, CEO, Novartis	CNBC Transcript: Vasant Narasimhan, CEO, Novar...
4	c2a81b0f93c4742de98785865cf855e7	https://www.cnbc.com/2018/03/26/fiscal-policy-...	Fiscal policy is 'spinning out of control': Ex...	Fiscal policy is 'spinning out of control': Ex...

(a) CODEC Dataset Corpus Example



	query_id	query	domain	guidelines
0	economics-1	How has the UK's Open Banking Regulation benef...	finance	UK's Open Banking regulation, which has parall...
1	economics-2	What technological challenges does Bitcoin fac...	finance	Bitcoin is a decentralized digital currency, w...
2	economics-3	Why are many commentators arguing NFTs are the...	finance	A non-fungible token (NFT) is a non-interchang...
3	economics-4	Why has value investing underperformed growth ...	finance	Value investing is the art of buying stocks th...
4	economics-6	Why are some economists sceptical about the EU...	finance	The euro arose from the 1991 Maastricht Treaty...

(b) CODEC Dataset Queries Example



	query_id	doc_id	relevance	iteration
0	economics-8	089a22846f6ba15fb4ef4cca0a884dd4	2	Q0
1	economics-8	0b2b173ef8348037ad80067ff37f0955	0	Q0
2	economics-8	0f78f8ec15ffec5ee7b8428af6b78683	0	Q0
3	economics-8	106e0f2b21616c3b30fd3b82127e4f5e	1	Q0
4	economics-8	143dc397c844340d13db74d4a34a78b8	1	Q0
...	...	...	...	...
6181	politics-23	bdebee3a5fd877a9ae7ffb80ca87d79f	1	Q0

(c) CODEC Dataset Qrels Example

Figure 3.3: Examples from the CODEC Dataset. (a) The corpus consists of complex, multi-faceted documents from domains such as economics and politics. (b) The queries represent intricate, context-dependent search queries used for evaluation. (c) The Qrels (query relevance judgments) provide mappings between queries and relevant corpus documents, assigning relevance scores for evaluation.

trieval results for complex, multifaceted queries. By evaluating retrieval systems using CODEC, researchers can better understand the strengths and limitations of current models and explore opportunities for improving **semantic search** techniques in **complex reasoning scenarios**.

The **CODEC** dataset is freely available for use, providing researchers with a valuable resource to evaluate information retrieval models in complex, high-impact societal domains such as **economics**, **history**, and **politics**. Researchers can utilize the dataset to conduct experiments focused on **entity-centric retrieval**, **semantic search**, and **complex reasoning scenarios**. The dataset's availability allows for extensive experimentation and benchmarking, enabling deeper insights into how retrieval models can handle **multi-faceted, contextual queries**.

The selected datasets contain both positive and negative labels. However, in this research, only positive examples are utilized for experimentation. Table 3.1 presents the unique number of elements in the corpus, queries, and relevance judgments (qrels) for each dataset in its initial version.

Table 3.1: Dataset Statistics Version 1

Model	**# unique corpus elements**	**# unique query elements**	**Positive # qrels**	**Negative # qrels**
nf-corpus	3593	3216	110575	0
fiqa	57599	6648	14166	0
codec	728752	42	3066	1882

As observed in Table 3.1, the **NF Corpus** dataset has significantly more positive qrels compared to the other two datasets. This imbalance could pose challenges during experimentation, as an unbalanced dataset might influence the evaluation results. To address this issue, the **NF Corpus** dataset was subsampled in the second version of the dataset. The updated statistics for version 2 are provided in Table 3.2.

Table 3.2: Dataset Statistics Version 2

Model	# unique corpus elements	# unique query elements	Positive # qrels	Negative # qrels
nf-corpus	3593	3216	10000	0
fiqa	57599	6648	14166	0
codec	728752	42	3066	1882

Additionally, as shown in both Table 3.1 and Table 3.2, the **CODEC** dataset exhibits a substantially lower number of unique queries compared to the number of unique corpus elements. This indicates that each query is associated with multiple relevant corpus elements, with some queries potentially having more than five relevant documents.

Furthermore, while **NF Corpus** and **FiQA** datasets primarily focus on a single domain, the **CODEC** dataset spans three distinct domains. The distribution of different domains within the **CODEC** dataset is presented in Table 3.5.

Table 3.3: #query Domain Distribution

query domain	count
economics	14
history	14
politics	14

Table 3.4: #corpus Domain Distribution

corpus domain	count
N/A	726028
economics	1289
history	972
politics	1448

Table 3.5: CODEC Dataset Domain Statistics

As shown in Table 3.5, the CODEC corpus includes entries labeled as *N/A* in the domain category. This indicates that certain relevance judgments (qrels) do not correspond to any specific query domain, meaning these entries are not associated with any particular query. In some cases, such instances are intentionally introduced to create noise in the corpus for train-

ing purposes. This practice can be beneficial for certain research-related experiments or model enhancement strategies.

As mentioned earlier, while there are 14 unique queries per domain, the corpus contains a significantly larger number of documents. The primary reason for this discrepancy is that a single query can be associated with multiple relevant passages or documents, depending on the dataset structure.

The **word and character count** of queries and documents plays a crucial role in semantic search and retrieval performance, particularly in contrastive learning-based models. Variations in text length can significantly influence how well embedding models capture semantic meaning and distinguish between relevant and non-relevant documents.

In **Information Retrieval (IR)** tasks, a common assumption is that queries tend to be shorter, representing concise user intents, while answers or relevant documents are longer, providing detailed information. This assumption stems from traditional search engine paradigms and question-answering tasks, where a brief query (e.g., *"What is inflation?"*) is expected to retrieve a more comprehensive answer in the form of a document, article, or structured response. However, while this assumption holds in many cases, its validity can depend on domain-specific retrieval needs and dataset characteristics.

Table 3.6: Corpus Word-Character Distribution Statistics

		Word Statistics		Character Statistics	
Dataset	**Domain**	**mean**	**sum**	**mean**	**sum**
CODEC	N/A	664.658	482561042	4087.631	2967735171
	finance	959.029	1236189	5964.093	7687717
	history	1480.729	1439269	9114.416	8859213
	politics	927.868	1343553	5785.409	8377273
FiQA	finance	132.904	7660348	768.210	44278135
NF	medical	233.765	849269	1590.783	5779318

The word and character count distribution appears appropriate for each dataset, as shown in Table 3.6. However, as observed in Table 3.7, which presents the distribution statistics of query data, there is an imbalance in distribution across datasets. This imbalance could pose challenges during training.

Table 3.7: Queries Word-Character Distribution Statistics

		Word Statistics		Character Statistics	
Dataset	**Domain**	**mean**	**sum**	**mean**	**sum**
CODEC	finance	13.000	182	83.928	1175
	history	12.642	177	75.428	1056
	politics	11.785	165	73.285	1026
FiQA	finance	10.822	71945	62.709	416895
NF	medical	3.318	10743	22.933	74235

In particular, when comparing the NF Corpus dataset with the FiQA dataset, there is a **30.56%** difference in query statistics. FiQA contains significantly more words and characters per query, leading to discrepancies in learning dynamics. The NF Corpus dataset may yield lower performance results, as the model might struggle to learn effectively from NF data, especially when combined with the FiQA dataset.

Similar trends were observed in preliminary results, reinforcing the decision to exclude the NF Corpus dataset from this research. The detailed impact of this decision on performance is further analyzed and presented in the **Results** chapter.

To ensure the effectiveness of **prefix-driven contrastive learning** in **semantic search**, FiQA was ultimately selected due to its **well-structured format, domain relevance, and retrieval complexity**. Unlike the other considered datasets, FiQA provides a **balanced and informative query-document structure**, allowing models to learn meaningful **semantic representations** in a domain where **precision and contextual understanding are critical**. The dataset's **rich financial terminology, diverse query types, and real-world applicability** make it particularly well-suited for evaluating **contrastive learning-based retrieval methods**. Additionally,

the **query and document length distribution** in FiQA aligns well with standard **information retrieval (IR) paradigms**, ensuring a **stable and reliable training process**. Given its combination of **domain specificity, structured relevance labels, and diverse linguistic patterns**, FiQA presents an **ideal testbed** for assessing the impact of **prefix-driven learning on retrieval performance**, offering insights that extend beyond finance to other **high-impact societal domains**.

3.2 Model Selection

In this research, the **BGE (Base Generative Embeddings)** and **E5 (Embedding-Based Retrieval)** models were selected as the core architectures for evaluating **prefix-driven contrastive learning** in **semantic search** due to their strong performance in embedding-based retrieval tasks and their capacity to generate high-quality representations for information retrieval. The **BGE** model, specifically designed for dense retrieval and ranking tasks, offers robust embeddings that effectively capture semantic similarities, making it highly suited for tasks requiring **precision and contextual understanding**. Given its optimization for retrieval-based applications, BGE is an excellent candidate for **financial domain search**, where nuanced understanding of context and relevance is essential.

To begin the experimentation process, the **BGE small** model was selected as an initial starting point, allowing for more manageable training and faster results, which are crucial for iterative experimentation and comparison. After evaluating the performance of the **BGE small** model, the research then extended to the **BGE base** model to observe any improvements in retrieval effectiveness with a larger, more powerful architecture. This provided an important baseline for understanding the scalability of the BGE model and its performance across different sizes.

After obtaining results with BGE, the **E5 model**—a state-of-the-art embedding model trained specifically on contrastive learning objectives—was introduced to see how it compared. **E5** has been demonstrated to provide excellent retrieval performance across various benchmark datasets, and its support for **instruction-based embedding generation** made it a natural fit for experimenting with domain-specific prefixes in the **financial domain**. Initially, the **E5 small** model was experimented with to assess its ability to handle semantic search in the domain at a more computationally efficient scale.

The decision to start with the **BGE small** and **E5 small** models was driven by the need for computational efficiency and quick feedback during the experimentation phase. Subsequently, the transition to **BGE base** model enabled a direct comparison of performance between differ-

ent model sizes, providing insights into the scalability and effectiveness of these models in high-impact domains like finance. This research ultimately compares these models' performance to evaluate how **prefix-driven contrastive learning** influences their ability to retrieve relevant information effectively in the **financial semantic search setting**, contributing to a deeper understanding of each model's strengths, limitations, and best-use scenarios.

4

Results

This section presents the results of the conducted experiments, evaluating the effectiveness of **prefix-driven contrastive learning** for **semantic search** in financial and high-impact domains. The experiments were designed to assess various pretrained models, including **BGE small, BGE base, and E5 small**, using the **FiQA dataset**. To quantify retrieval effectiveness, multiple evaluation metrics were employed, such as **Mean Reciprocal Rank (MRR)** and **Normalized Discounted Cumulative Gain (NDCG)**.

All experiments were conducted under consistent training conditions to ensure fair comparisons across models. The fine-tuning process used a learning rate of **1e-5** with a **linear** scheduler, a warmup ratio of **0.9**, and a total of **5 epochs**. The batch size was set to **64** for BGE-small and E5-small, but due to memory limitations, **BGE-base was fine-tuned with a batch size of 32** to prevent CUDA out-of-memory errors. Training checkpoints were saved at **10 evenly spaced intervals** throughout training, and the maximum number of training steps was determined dynamically based on the dataset size.

All experiments were conducted using **Google Colab**, with a **T4 GPU** as the primary accelerator. This setup ensured a controlled computational environment, allowing for a direct comparison of different model architectures and training strategies.

The section begins with a description of the testing methodology, followed by an analysis of preliminary results and a comparative evaluation of different model configurations. Additionally, an error analysis is conducted to highlight challenges encountered during training and testing. The findings provide insights into the impact of prefix-driven contrastive learn-

ing and model selection on retrieval performance, ultimately contributing to best practices for fine-tuning pretrained models in specialized domains.

4.1 Testing Methodology

The testing methodology plays a crucial role in evaluating the performance of fine-tuned models. The **IR Evaluator** (Information Retrieval Evaluator) from the **Sentence Transformers** library is selected as the primary evaluation method. The IR Evaluator retrieves the **top-k most similar documents** for each query, allowing flexible selection of evaluation metrics based on experimental requirements.

Initially, the **NanoBeIR Evaluator**, a subsampled version of the BEIR benchmark, was considered for evaluation. However, even when testing the base model, the expected improvements were not observed. This discrepancy may stem from the significantly smaller dataset used in NanoBeIR, which could reduce the reliability and robustness of evaluation results. As a result, the IR Evaluator was chosen for its ability to leverage the **complete test split** of the dataset, ensuring a more **comprehensive and reliable** assessment of model performance.

To enable a thorough performance analysis, multiple evaluation metrics—including **MRR, accuracy, NDCG, recall, and mean average precision (MAP)**—are computed for $k = \{1, 5, 10, 15, 20, 25, 30\}$. This diverse set of metrics allows for a nuanced comparison of retrieval effectiveness, particularly in cases where models exhibit unexpected variations in performance.

Among these metrics, **MRR@5** is selected as the primary evaluation metric due to its effectiveness in assessing retrieval performance in **semantic search tasks**, particularly in **societal domain applications**. The **Mean Reciprocal Rank (MRR)** evaluates how quickly a relevant document appears in ranked retrieval results, making it especially suitable for tasks where retrieving **at least one highly relevant result early** is critical. By choosing $k = 5$, **MRR@5 directly measures the model's ability to surface relevant documents within the top five retrieved results**, which aligns with real-world search behavior.

In societal domains, such as those covered by the **FiQA dataset**, users typically seek a **single highly relevant answer** rather than a broad set of partially relevant documents. MRR@5 is well-suited for this scenario because it prioritizes **first-hit accuracy**, making it **more interpretable and actionable** compared to metrics like Recall@k or NDCG, which consider multiple relevance levels. Unlike recall-based metrics, which assess whether **all** relevant documents are retrieved, **MRR focuses on ranking efficiency**, ensuring that the most relevant information appears **early in the search results**. This characteristic is particularly important in societal

domains, where access to **immediate and accurate information** is crucial.

Furthermore, MRR@5 is computationally efficient and provides a **stable, high-resolution signal for ranking effectiveness**. Compared to broader recall-based metrics, which may include **distantly ranked relevant documents**, MRR@5 ensures that the emphasis remains on **retrieving the most relevant answer as quickly as possible**. This makes it an ideal choice for evaluating **prefix-driven contrastive learning**, where the goal is to optimize ranking performance within the **top few results**.

During the testing phase, **prefix settings** were systematically applied to both queries and corpus documents, maintaining consistency with training conditions. This ensured that the impact of **prefix-driven contrastive learning** was assessed under **controlled and replicable** conditions. By aligning training and testing settings, the evaluation method provides a **fair and reliable** comparison of model effectiveness across different **prefixing strategies**. This consistency is essential for **validating the efficacy of prefix-driven methods** in retrieval tasks and ensuring that observed improvements in performance **result from the applied prefix settings rather than external factors**.

4.2 Dataset Performance Analysis

As explained in Section 3.1, although the initial plan involved using three datasets for training, the experiments ultimately proceeded with only the FiQA dataset. The CODEC corpus was excluded due to its large size, which posed challenges for efficient and timely training. Meanwhile, the NF corpus was removed because it negatively impacted the performance of the fine-tuned model.

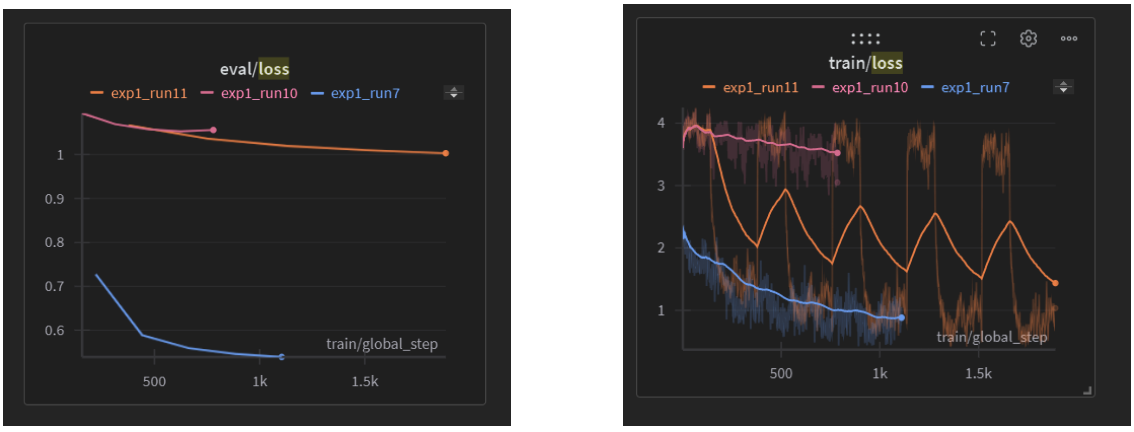
Throughout the training process, evaluations and training statistics were logged using Weights & Biases (wandb) to systematically track improvements and ensure a professional and transparent analysis. Initially, the BGE-small model was trained on a combined dataset consisting of FiQA and NF corpus, and its performance was evaluated on their test split. However, expected improvements were not observed in the early experiments. Separate fine-tuning was conducted on each dataset to investigate the cause of this performance degradation. The evaluation results demonstrated a performance ranking, with FiQA yielding the highest scores, followed by the combined FiQA + NF dataset, and finally the NF dataset alone. This ranking, based on MRR@5 scores, is presented in Table 4.1, further confirming that the NF corpus contributed to decreased model performance.

Table 4.1: MRR@5 Evaluation Results for NF Corpus, FiQA, and FiQA + NF Corpus Trained with BGE-Small

Model / Datasets	NF Corpus	FiQA	FiQA + NF Corpus
BGE-Small	0.538	0.752	0.677

The wandb logs provided a visualization of evaluation loss and training loss, as shown in Figure 4.1. The model fine-tuned solely on the FiQA dataset (`exp1_run7`) exhibited well-behaved loss curves, converging towards zero for both training and evaluation loss. In contrast, the model fine-tuned on the NF dataset alone (`exp1_run10`) initially showed a decline in loss but then began to increase, indicating instability. The model fine-tuned on the combined FiQA + NF dataset (`exp1_run11`) displayed loss values between the two, yet still remained relatively high without sufficient convergence to zero. These findings indicate that the NF dataset did not provide enough useful signal for the model to learn effectively.

Consequently, to ensure a clearer evaluation of prefix-driven contrastive learning, the NF dataset was eliminated from further experiments, and the research continued exclusively with the FiQA dataset.



(a) Eval/Loss **(b)** Train/Loss

Figure 4.1: Evaluation and Train Loss for NF Corpus, FiQA, and FiQA + NF Corpus Trained with BGE-Small

4.3 Baseline Model Comparison on Test Method

In the testing phase, fine-tuned models were evaluated to assess their retrieval performance using the established test methodology. Additionally, baseline models—**BGE-Small, BGE-**

Base, and E5-Small—were tested on the **FiQA dataset** using the same evaluation procedure. This ensured a consistent comparison between fine-tuned models and their pretrained counterparts, providing insights into the effectiveness of prefix-driven contrastive learning.

Each experiment setting's corresponding base model was evaluated under identical test conditions. Specifically, for **BGE-Small**, four different prefix configurations were tested:

- **W/O Prefix**: without any prefix modification,
- **W/ BGE Query Prefix**: prefix added only to the query,
- **W/ BGE Query Domain-Specific Prefix**: domain-specific prefix applied to the query,
- **W/ BGE Query and Passage Domain-Specific Prefix**: prefix applied to both the query and the passage.

For **BGE-Base**, due to efficiency and computational constraints, only **W/O Prefix** and **W/ BGE Prefix** configurations were explored. This decision was made to balance training time and performance evaluation while still capturing key trends in retrieval performance.

For **E5-Small**, the same experimental settings used in **BGE-Small** were applied, with the addition of the **W/ E5 Prefix** setting. This configuration was included because **E5 is pretrained with an inherent prefix structure**, where `"query:"` and `"passage:"` prefixes are applied to queries and documents, respectively. By explicitly testing this format, the experiment aimed to analyze whether leveraging the **E5-specific prefix structure** would enhance retrieval performance in the financial domain.

The **MRR@5 scores** of all base models across different prefix settings can be observed in Table 4.3. These results offer a comparative perspective on how different models and prefix strategies impact retrieval effectiveness. Moreover, they provide evidence on whether **prefix-driven contrastive learning** improves performance relative to standard dense retrieval approaches.

4.4 Experiment Results

This section presents and analyzes the results of the conducted experiments, providing a detailed evaluation of the findings. The experimental outcomes are systematically discussed, with relevant tables and figures illustrating key observations. The primary hypothesis of this research posits that incorporating more complex prefix structures will enhance the performance of a domain-specific fine-tuned model. This assumption is based on the premise that richer

Table 4.2: Experiment settings for different prefix variations used in retrieval tasks. Each experiment applies a unique prefix configuration to queries and corpus passages, impacting the model's understanding and performance.

Experiment	Prefix Details
W/O Prefix	**Query:** None, **Corpus:** None
W/ E5 Prefix	**Query:** 'query:', **Corpus:**'passage:'
W/ BGE Prefix	**Query:** 'Represent this sentence for searching relevant passages:', **Corpus:** None
W/ BGE Domain Prefix	**Query:** 'Represent this sentence for searching relevant passages about DOMAIN:', **Corpus:** None
W/ BGE Query & Passage Domain Prefix	**Query:** 'Represent this sentence for searching relevant passages about DOMAIN:', **Corpus:** 'Represent this passage about DOMAIN to make it searchable:'

prefix-based contextualization can provide additional semantic guidance, thereby improving retrieval effectiveness in specialized domains such as finance.

The results of the experiments offer partial validation of this hypothesis, demonstrating performance improvements in certain configurations. However, some unexpected variations in model behavior were observed, necessitating a deeper investigation into the underlying factors contributing to these deviations. These findings highlight the complexity of prefix-driven contrastive learning and its interaction with domain-specific training. Consequently, the following sections provide a comprehensive breakdown of experimental outcomes, analyzing both the expected and unexpected results in order to derive meaningful insights and refine the application of prefix strategies in semantic search. All the results can be observed in Table 4.3.

Table 4.3: MRR@5 evaluation results for baseline and fine-tuned models on the FiQA dataset across different prefix settings. The table compares the impact of various prefix configurations on retrieval performance.

Model / Settings	W/O Prefix	W/ E5 Prefix	W/ BGE Prefix	W/ BGE Domain Prefix	W/ BGE Query&Passage Domain Prefix
Baseline BGE-Small	0.721	N/A	0.731	0.731	0.715
Baseline BGE-Base	0.747	N/A	0.754	N/A	N/A
Baseline E5 Small	0.692	0.708	0.633	0.615	0.542
BGE-Small Fine-Tuned	0.752	N/A	0.754	0.750	0.756
BGE-Base Fine-Tuned	0.775	N/A	0.773	N/A	N/A
E5 Small Fine-Tuned	0.759	0.760	0.762	0.759	0.716

4.4.1 Performance Gains from Fine-Tuning: A Comparison with Base Models

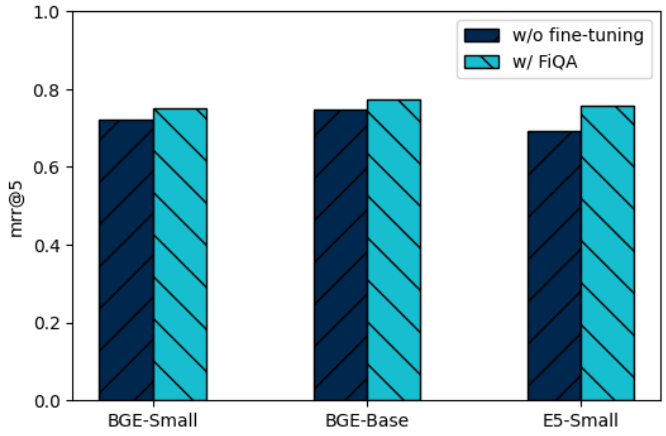


Figure 4.2: Impact of Fine-Tuning with a Domain-Specific Dataset on Retrieval Performance

Fine-tuning pretrained models on domain-specific data is expected to improve retrieval performance by allowing the model to learn task-specific representations. To evaluate this claim, the BGE-Small, BGE-Base, and E5-Small models were fine-tuned without prefixes using the FiQA dataset, and their performance was compared against their respective base models. The evaluation was conducted using MRR@5 to measure how effectively the fine-tuned models ranked relevant documents.

The results confirm that fine-tuning significantly enhances retrieval performance across all tested models and can be observed from Figure 4.2. Among them, the E5-Small model exhibited the highest improvement, followed by BGE-Base, and lastly BGE-Small. This trend aligns with expectations due to differences in model architecture, pretraining strategy, and capacity. E5-Small, which was pretrained using contrastive learning and instruction-based retrieval, led to the most significant performance boost. BGE-Base, benefiting from its larger size, captures more nuanced domain-specific patterns after fine-tuning, showing moderate improvement. In contrast, BGE-Small, with its limited representational capacity, demonstrated the least performance gain. The results of fine-tuned model is presented on Table4.3.

These findings validate the hypothesis that fine-tuning enhances retrieval performance by aligning model representations. However, the extent of improvement varies based on the model's pretraining approach and parameter capacity, emphasizing the importance of selecting an appropriate base model for fine-tuning in specialized domains like finance.

4.4.2 Impact of Basic Query Prefixes on Model Performance

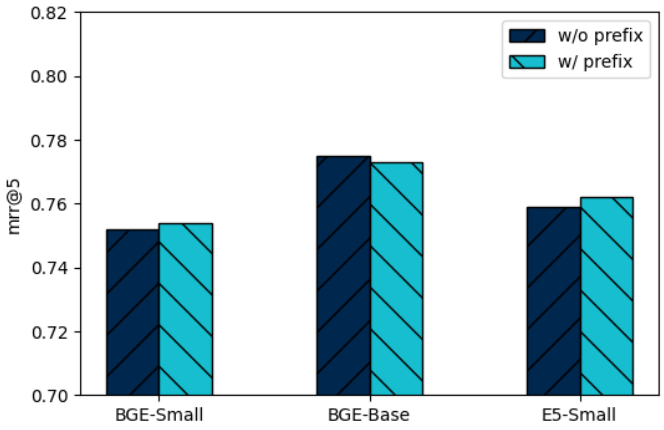


Figure 4.3: Performance Improvement of Fine-Tuned Models with Query Prefixes Compared to Fine-Tuning Without Prefixes

One of the central objectives of this research is to investigate whether incorporating basic prefixes during fine-tuning enhances the retrieval performance of the model. To evaluate this claim, a comparative analysis was conducted between three key configurations: (1) the base (pretrained) model, (2) the fine-tuned model without prefixes (W/O Prefix), and (3) the fine-tuned model with a query prefix (W/ Query Prefix). By systematically comparing these variations, this study aims to assess the impact of prefix-driven contrastive learning on retrieval effectiveness in the financial domain.

For the **BGE-Small** and **BGE-Base** models, the prefix `"Represent this sentence for searching relevant passages:"`—referred to as the **BGE Query Prefix**—was used during fine-tuning. This prefix is designed to guide the model toward learning representations optimized for retrieval tasks, ensuring that query embeddings align effectively with relevant passage embeddings. Since the **E5 model** is pretrained using an instruction-based embedding strategy, an additional prefix configuration was explored. Specifically, two distinct prefixes were used for the E5 model: (1) the **BGE Query Prefix**, identical to the one used for BGE models, and (2) the **E5 native prefixes**, where `"query:"` was appended to queries and `"passage:"` was prepended to corpus entries. The rationale behind including E5's native prefixes is that this model was originally trained with them, potentially allowing for better alignment with its pretraining objectives and leading to superior retrieval performance.

The decision to experiment with multiple prefix strategies stems from the hypothesis that **prefix-driven contrastive learning can help models learn richer semantic representations**

by explicitly distinguishing between queries and passages. Given that semantic search tasks in the financial domain require high precision and context awareness, the introduction of query prefixes is expected to improve ranking quality by reinforcing the distinction between user intent (queries) and relevant information (passages). The details of each prefix formulation and its function in the retrieval process are further outlined in Table **??**.

In the results presented in the table, it is evident that adding a basic query prefix leads to a noticeable improvement in the performance of most models tested. Specifically, for the fine-tuned **E5 Small** models, performance improvements are observed when using both the **E5 prefix** and the **BGE Query prefix**. Notably, the **BGE Query prefix** resulted in an even more substantial performance boost compared to the E5 prefix. This observation supports the hypothesis that incorporating query prefixes during fine-tuning can enhance model performance, particularly by guiding the model to better distinguish between query intent and relevant passages.

In addition to Table 4.3, Figure 4.3 visually illustrates the performance improvements achieved by incorporating query prefixes during fine-tuning. The plot highlights the relative gains across different models, further supporting the observed trends.

However, one notable exception to this trend was observed in the **BGE-Base** model, where the addition of the query prefix resulted in a slight performance decrease (from 0.775 to 0.773). This minor drop could be attributed to the fact that BGE-Base may already be highly optimized for retrieval tasks without the need for a query prefix. Introducing a prefix alters the tokenization of inputs and may shift the embedding distribution in a way that does not necessarily benefit performance. Additionally, fine-tuning with prefixes can introduce small variations in optimization dynamics, which might not always lead to a net improvement, particularly if the dataset does not strongly benefit from additional contextual cues.

These findings serve as a first step in validating the claim that fine-tuning with the addition of basic query prefixes improves retrieval performance. While the overall trend supports the effectiveness of prefixes in retrieval tasks, the case of BGE-Base highlights the importance of considering model-specific behaviors and the potential limitations of prefix-driven fine-tuning. The observed improvements underline the significance of prefix-driven contrastive learning in refining model representations for more accurate and efficient semantic search, particularly in specialized domains like finance.

4.4.3 Impact of More Complex Prefixes on Model Performance

The models were first fine-tuned in the more advanced prefix experiments using a **domain-specific query prefix**. However, the results did not align with expectations, as this configuration led to a decrease in performance compared to the basic query prefix experiment. One possible explanation for this outcome is that the model, being fine-tuned on a single-domain dataset, was unable to effectively leverage the domain-specific prefixes. Since all training data originated from the same domain, the model may not have gained enough variation in query and passage structures to learn effectively from the added prefix. This could have limited the model's ability to generalize well to new queries, as it may have become overly accustomed to the specific patterns of the dataset without expanding its understanding of how the domain-specific prefix interacts with more varied queries.

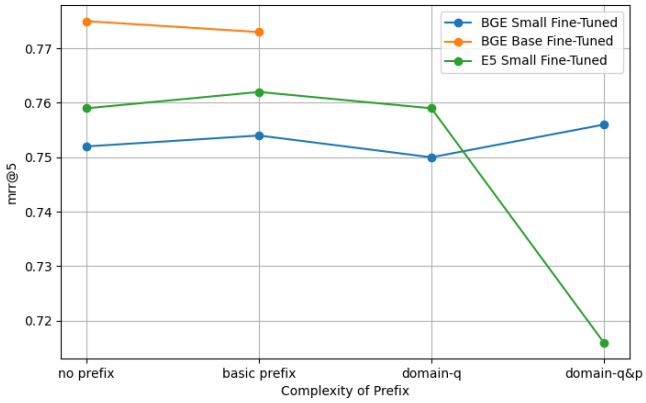


Figure 4.4: Performance Comparison of Fine-Tuned Models with Varying Prefix Complexities. The graph illustrates the Mean Reciprocal Rank at rank 5 (MRR@5) for three fine-tuned models (BGE-Small, BGE-Base, and E5 Small) across different prefix configurations, ranging from basic query prefixes to more complex domain-specific query and passage prefixes. The results highlight the impact of prefix complexity on retrieval performance, with performance variations observed between each model's setup. Note that the results for the E5 prefix configuration are not included in this figure, as it was only applied to the E5 Small model.

For the **E5 Small** model, the domain-specific query prefix yielded even lower performance than the setup using the standard **E5 prefix**. This suggests that, despite E5's pretraining with instruction-based prompts, the introduction of a highly specialized prefix did not enhance retrieval effectiveness. Instead, it may have restricted the model's ability to generalize beyond the narrow patterns present in the training dataset, which could also imply that the model did not have enough exposure to diverse query structures or broader data distribution during training.

To address these limitations, a second, more complex prefix configuration was introduced: the **Query and Passage Domain-Specific Prefix**. This setup incorporated domain-specific prefixes not only in the query but also in the passage representations. The results demonstrated a noticeable shift in performance. For the **BGE-Small** fine-tuned model, this configuration yielded the **highest performance** across all experiments, indicating that the simultaneous enrichment of both queries and passages with domain-specific context allowed the model to develop a more effective semantic representation. In contrast, the **E5 Small** model still exhibited lower performance under this configuration, reinforcing that model architecture and pretraining objectives play a crucial role in how prefixes influence retrieval effectiveness.

These findings suggest that while incorporating more complex prefixes in both queries and passages can enhance model learning, the effectiveness of this approach is highly dependent on the model's architecture and pretraining strategy. The improved performance of BGE-Small with this setup indicates that prefix-driven contrastive learning can be beneficial when carefully aligned with the model's characteristics and the dataset's complexity. Furthermore, these results highlight the necessity of tailoring prefix configurations based on the underlying retrieval model, as different architectures may respond differently to various prefix strategies. Future research could explore the impact of additional training data diversity or multi-domain datasets to assess whether broader exposure to varied prefix structures can mitigate the limitations observed in fine-tuning on a single-domain dataset, which may restrict the model's ability to fully benefit from more complex prefixes.

In this section, we presented and analyzed the results of various experiments conducted to evaluate the performance of fine-tuned models for semantic search in specialized domains. The findings demonstrate that fine-tuning with domain-specific datasets leads to significant improvements in retrieval performance when compared to base models. Furthermore, the introduction of basic prefixes (e.g., BGE query prefix) during fine-tuning proved to enhance model performance, confirming our hypothesis that incorporating prefixes improves retrieval accuracy.

However, some results were not as expected, particularly when using more complex domain-specific prefixes. In some cases, the models showed lower performance, suggesting that more intricate prefixes may lead to insufficient learning from the dataset or prefix. Despite these challenges, the results indicate that the choice of prefix configuration has a notable impact on retrieval quality, and the selection of the appropriate prefix setup depends on the model and dataset characteristics.

These findings have important implications for improving model performance in domain-

specific applications such as financial semantic search, where the accuracy of retrieved documents is crucial. While the results were promising, further exploration is necessary to refine the prefix configurations and address the challenges identified.

5
Conclusion

This thesis investigates the impact of fine-tuning domain-specific models using prefix-driven contrastive learning techniques for semantic search, with a particular focus on the financial and high-impact domains. [43] The primary goal of this research was to evaluate the effectiveness of different pretrained models—namely, BGE-Small, BGE-Base, and E5—fine-tuned with a domain-specific dataset (FiQA). Several key findings were obtained through rigorous experimentation, which involved testing models with and without prefixes, as well as exploring different prefix configurations such as basic and domain-specific prefixes. The evaluation results demonstrated that fine-tuning with domain-specific data significantly improved model performance, as measured by MRR@5, compared to the base models. This improvement was particularly evident in the E5 model, which showed the most significant gain due to its pretraining with contrastive learning methods that aligned well with the semantic search task. Furthermore, the addition of simple query prefixes further boosted performance, especially when applied to the E5-Small model. However, more complex prefixes, including both query and passage domain-specific prefixes, led to unexpected performance drops for certain models, notably for the E5 model. This suggests that the model might have been overfitting when exposed to a narrow set of domain-specific data, highlighting the importance of balancing the complexity of the prefix setup with the domain-specific learning.

Despite the promising improvements observed with basic prefixes, the research also faced challenges, particularly related to dataset selection. The inclusion of additional datasets such as NF corpus led to performance degradation, emphasizing the importance of dataset quality and

domain relevance when training models for semantic search tasks. The findings indicate that models such as E5 benefit more from fine-tuning with a coherent and representative dataset like FiQA, rather than larger, less focused datasets.

While the results support the claim that domain-specific fine-tuning, coupled with appropriate prefix usage, can significantly enhance retrieval performance in semantic search tasks, this research also uncovered areas for improvement and refinement. For instance, testing more complex prefix setups revealed a potential overfitting problem, suggesting that further research could explore more refined strategies to prevent overfitting and optimize prefix configurations.

6 Limitations and Future Works

6.1 Limitations

This study has a few limitations that should be acknowledged. First, the experiments were carried out using a single domain-specific dataset (FiQA) and a relatively limited set of model architectures. While these models showed promising results, their effectiveness may vary when applied to other domains or larger datasets. Second, the performance of complex prefix configurations was not always as expected, indicating that more experimentation is needed to understand the interplay between domain-specific fine-tuning and prefix complexity.

6.2 Future Works

The findings from this research lay the groundwork for further exploration in the area of **prefix-driven contrastive learning** for semantic search. Several promising avenues for future work include:

- **Expanding Prefix Configurations**: Future experiments could investigate more complex combinations of query and passage prefixes to better understand their impact on model performance. This includes testing prefixes beyond the basic and domain-specific setups, exploring hierarchical or multi-layered prefix structures to further fine-tune models.

- **Larger and More Complex Datasets**: The **FiQA** dataset used in this research offers valuable insights, but it represents only a single domain (finance). Future work could explore the use of larger, multi-domain datasets to further test the generalization capabilities of the models. This could involve leveraging datasets from various industries or domains to test the adaptability and robustness of the model across different contexts.
- **Multilingual Models**: Another interesting direction for future research is testing the prefix-driven contrastive learning approach on multilingual datasets. Semantic search is a critical task in many languages, and testing the models' performance across different linguistic structures could provide insights into their scalability and performance in non-English domains.
- **Exploring Larger Models**: The models used in this study, such as **BGE-Small**, **BGE-Base**, and **E5-Small**, are relatively smaller versions, and their capacity may limit their performance on large-scale, high-complexity tasks. Future work could investigate the use of larger models (e.g., **BERT-large**, **T5**, **GPT-based architectures**) to explore whether increased model size improves the efficacy of prefix-based fine-tuning techniques.
- **Real-World Applications**: Finally, the real-world deployment of these models for **semantic search** tasks in specialized domains could provide valuable insights. By integrating the fine-tuned models into search engines or knowledge retrieval systems, we could better understand their practical performance and scalability in real-time settings, as well as their ability to handle user queries in complex, real-world scenarios.

In summary, while this study successfully demonstrated the advantages of domain-specific fine-tuning and prefix-driven contrastive learning for semantic search, it also opened up new areas for exploration. By refining the use of prefixes, expanding the dataset scope, and testing with more advanced models, future research can push the boundaries of **semantic search** and improve the efficacy of **information retrieval systems** in various specialized domains.

References

[1] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. Kaiser, and I. Polosukhin, "Attention is all you need," 2023. [Online]. Available: https://arxiv.org/abs/1706.03762

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," 2019. [Online]. Available: https://arxiv.org/abs/1810.04805

[3] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," 2019. [Online]. Available: https://arxiv.org/abs/1908.10084

[4] C. D. Manning, P. Raghavan, and H. Schütze, *Introduction to Information Retrieval*. Cambridge, England: Cambridge University Press, 2008. [Online]. Available: https://nlp.stanford.edu/IR-book/

[5] B. Mitra and N. Craswell, "An introduction to neural information retrieval," *Foundations and Trends® in Information Retrieval*, vol. 13, no. 1, pp. 1–126, 2018. [Online]. Available: http://dx.doi.org/10.1561/1500000061

[6] L. Badia, V. Bonagura, F. Pascucci, V. Vadori, and E. Grisan, "Medical self-reporting with adversarial data injection modeled via game theory," in *Proceedings of the IEEE International Conference on Communications, Signal Processing and Applications (ICCSPA)*, Publisher Location, 2024, pp. 1–6.

[7] P. Borlund, "The concept of relevance in IR," *Journal of the American Society for Information Science and Technology*, vol. 54, no. 10, pp. 913–925, 2003.

[8] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych, "Beir: A heterogenous benchmark for zero-shot evaluation of information retrieval models," 2021. [Online]. Available: https://arxiv.org/abs/2104.08663

[9] O. Chapelle, D. Metlzer, Y. Zhang, and P. Grinspan, "Expected reciprocal rank for graded relevance," in *CIKM '09: Proceeding of the 18th ACM conference on Information*

and knowledge management, New York, NY, USA, 2009, pp. 621–630. [Online]. Available: http://doi.acm.org/10.1145/1645953.1646033

[10] F. Radlinski, M. Kurup, and T. Joachims, "How does clickthrough data reflect retrieval quality?" in *Proceedings of the 17th ACM Conference on Information and Knowledge Management*, ser. CIKM '08. New York, NY, USA: Association for Computing Machinery, 2008, p. 43–52. [Online]. Available: https://doi.org/10.1145/1458082.1458092

[11] R. Nogueira and K. Cho, "Passage re-ranking with bert," 2020. [Online]. Available: https://arxiv.org/abs/1901.04085

[12] D. Scapin, G. Cisotto, E. Gindullina, and L. Badia, "Shapley value as an aid to biomedical machine learning: a heart disease dataset analysis," in *2022 22nd IEEE International Symposium on Cluster, Cloud and Internet Computing (CCGrid)*, 2022, pp. 933–939.

[13] Google AI, "Bert for search: Replacing keyword matching with semantic understanding," 2020. [Online]. Available: https://blog.google/products/search/search-language-understanding-bert/

[14] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent dirichlet allocation," *Journal of Machine Learning Research*, vol. 3, pp. 993–1022, 2003. [Online]. Available: https://www.jmlr.org/papers/volume3/blei03a/blei03a.pdf

[15] W. B. Croft, D. Metzler, and T. Strohman, *Search Engines: Information Retrieval in Practice*. Pearson, 2010, covers SVMs, KNN, and decision trees for document classification/ranking.

[16] Google AI, "T5 for document retrieval in google search," 2021. [Online]. Available: https://ai.googleblog.com/2021/10/t5x-new-research-framework-for-high.html

[17] J. Teevan, S. T. Dumais, and E. Horvitz, "Personalizing search via automated analysis of interests and activities," *SIGIR*, pp. 449–456, 2005.

[18] A. Zancanaro, G. Cisotto, and L. Badia, "Tackling age of information in access policies for sensing ecosystems," *Sensors*, vol. 23, no. 7, 2023. [Online]. Available: https://www.mdpi.com/1424-8220/23/7/3456

[19] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," in *EMNLP*, 2020, pp. 6769–6781.

[20] S. Rendle, C. Freudenthaler, Z. Gantner, and L. Schmidt-Thieme, "Bpr: Bayesian personalized ranking from implicit feedback," *UAI*, pp. 452–461, 2012. [Online]. Available: https://arxiv.org/abs/1205.2618

[21] Google AI, "How bert advances search understanding," 2020. [Online]. Available: https://blog.google/products/search/search-language-understanding-bert/

[22] J. Johnson, M. Douze, and H. Jégou, "Billion-scale similarity search with gpus," 2017. [Online]. Available: https://arxiv.org/abs/1702.08734

[23] O. Chapelle and Y. Chang, "Yahoo! learning to rank challenge overview," in *Proceedings of the Learning to Rank Challenge*, ser. Proceedings of Machine Learning Research, O. Chapelle, Y. Chang, and T.-Y. Liu, Eds., vol. 14. Haifa, Israel: PMLR, 25 Jun 2011, pp. 1–24. [Online]. Available: https://proceedings.mlr.press/v14/chapelle11a.html

[24] S. Barocas, M. Hardt, and A. Narayanan, *Fairness and Machine Learning: Limitations and Opportunities*. MIT Press, 2023. [Online]. Available: https://fairmlbook.org/

[25] V. Sanh, A. Webson, C. Raffel, S. H. Bach, L. Sutawika, Z. Alyafeai, A. Chaffin, A. Stiegler, T. L. Scao, A. Raja, M. Dey, M. S. Bari, C. Xu, U. Thakker, S. S. Sharma, E. Szczechla, T. Kim, G. Chhablani, N. Nayak, D. Datta, J. Chang, M. T.-J. Jiang, H. Wang, M. Manica, S. Shen, Z. X. Yong, H. Pandey, R. Bawden, T. Wang, T. Neeraj, J. Rozen, A. Sharma, A. Santilli, T. Fevry, J. A. Fries, R. Teehan, T. Bers, S. Biderman, L. Gao, T. Wolf, and A. M. Rush, "Multitask prompted training enables zero-shot task generalization," 2022. [Online]. Available: https://arxiv.org/abs/2110.08207

[26] A. Buratto, M. Levorato, and L. Badia, "Dcp: a tcp-inspired method for online domain adaptation under dynamic data drift," in *2024 IEEE 25th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 2024, pp. 269–278.

[27] I. Sutskever, O. Vinyals, and Q. V. Le, "Sequence to sequence learning with neural networks," in *Advances in Neural Information Processing Systems (NeurIPS)*, 2014, pp. 3104–3112.

[28] K. Cho, B. Van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio, "Learning phrase representations using rnn encoder-decoder for statistical machine translation," *arXiv preprint arXiv:1406.1078*, 2014. [Online]. Available: https://arxiv.org/abs/1406.1078

[29] D. Bahdanau, K. Cho, and Y. Bengio, "Neural machine translation by jointly learning to align and translate," *arXiv preprint arXiv:1409.0473*, 2014. [Online]. Available: https://arxiv.org/abs/1409.0473

[30] L. Wang, N. Yang, X. Huang, B. Jiao, L. Yang, D. Jiang, R. Majumder, and F. Wei, "Text embeddings by weakly-supervised contrastive pre-training," 2024. [Online]. Available: https://arxiv.org/abs/2212.03533

[31] L. Wang, N. Yang, X. Huang, L. Yang, R. Majumder, and F. Wei, "Multilingual e5 text embeddings: A technical report," 2024. [Online]. Available: https://arxiv.org/abs/2402.05672

[32] IntFloat, "E5 model," https://huggingface.co/intfloat, 2025, accessed: 2025-02-26.

[33] S. Xiao, Z. Liu, P. Zhang, N. Muennighoff, D. Lian, and J.-Y. Nie, "C-pack: Packed resources for general chinese embeddings," 2024. [Online]. Available: https://arxiv.org/abs/2309.07597

[34] B. B. A. of Artificial Intelligence), "Bge: Bilingual general embeddings," 2023, accessed: 2023-02-26. [Online]. Available: https://huggingface.co/BAAI

[35] Pinecone, "Offline evaluation for information retrieval," 2023, accessed: 2025-02-26. [Online]. Available: https://www.pinecone.io/learn/offline-evaluation/

[36] N. Thakur and N. Reimers, "Fiqa dataset on hugging face (beir collection)," 2021. [Online]. Available: https://huggingface.co/datasets/BeIR/fiqa

[37] H. Face, "Nf corpus dataset," 2025. [Online]. Available: https://huggingface.co/datasets/BeIR/nfcorpus/viewer/queries?views%5B%5D=queries

[38] ——, "Codec dataset," 2025. [Online]. Available: https://huggingface.co/datasets/irds/codec

[39] F. D. Authors, "Fiqa: Financial question answering dataset," 2018. [Online]. Available: https://sites.google.com/view/fiqa/

[40] H. Face, "MTEB leaderboard," 2025. [Online]. Available: https://huggingface.co/spaces/mteb/leaderboard

[41] U. of Heidelberg StatNLP Group, "Nf corpus," 2025. [Online]. Available: https://www.cl.uni-heidelberg.de/statnlpgroup/nfcorpus/

[42] G. Lab, "Codec github repository," 2025, accessed: 2025-03-27. [Online]. Available: https://github.com/grill-lab/CODEC

[43] T. Erseghe, L. Badia, L. Dzanko, and C. Suitner, "Plmp: A method to map the linguistic markers of the social discourse onto its semantic network," in *2022 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM)*, 2022, pp. 247–251.

Acknowledgments