NEURAL INFORMATION RETRIEVAL

Project Submitted to the
SRM University AP, Andhra Pradesh
for the partial fulfillment of the requirements to award the degree of

Bachelor of Technology
in
Computer Science & Engineering
School of Engineering & Sciences

submitted by

Lovely Yeswanth Panchumarthi(AP20110010299)

Sriya Padmanabhuni(AP20110010274)

Lavanya Parchuri(AP20110010233)
Yogeshvar Reddy Kallam(AP20110010145)

Under the Guidance of

Dr. Satya Krishna Nunna

Department of Computer Science & Engineering
SRM University-AP
Neerukonda, Mangalgiri, Guntur
Andhra Pradesh - 522 240
May 2024

DECLARATION

I undersigned hereby declare that the project report **Neural Information Retrieval** submitted for partial fulfillment of the requirements for the award of degree of Bachelor of Technology in the Computer Science & Engineering, SRM University-AP, is a bonafide work done by me under supervision of Dr. Satya Krishna Nunna. This submission represents my ideas in my own words and where ideas or words of others have been included, I have adequately and accurately cited and referenced the original sources. I also declare that I have adhered to ethics of academic honesty and integrity and have not misrepresented or fabricated any data or idea or fact or source in my submission. I understand that any violation of the above will be a cause for disciplinary action by the institute and/or the University and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been obtained. This report has not been previously formed the basis for the award of any degree of any other University.

Place	:	Date	: May 14, 2024
Name of student	: Lovely Yeswanth Panchumarthi	Signature	: .....
Name of student	: Sriya Padmanabhuni	Signature	: .....
Name of student	: Lavanya Parchuri	Signature	: .....
Name of student	: Yogeshvar Reddy Kallam	Signature	: .....

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

SRM University-AP

Neerukonda, Mangalgiri, Guntur

Andhra Pradesh - 522 240

CERTIFICATE

This is to certify that the report entitled **Neural Information Retrieval** submitted by **Lovely Yeswanth Panchumarthi, Sriya Padmanabhuni, Lavanya Parchuri, Yogeshvar Reddy Kallam** to the SRM University-AP in partial fulfilment of the requirements for the award of the Degree of Bachelors of Technology in Computer Science & Engineering is a bonafide record of the project work carried out under my/our guidance and supervision. This report in any form has not been submitted to any other University or Institute for any purpose.

Project Guide	Head of Department
Name : Dr. Satya Krishna Nunna	Name : Dr. Niraj Upadhayaya
Signature:	Signature: .....

ACKNOWLEDGMENT

I wish to record my indebtedness and thankfulness to all who helped me prepare this Project Report titled **Neural Information Retrieval** and present it satisfactorily.

I am especially thankful for my guide and supervisor Dr. Satya Krishna Nunna in the Department of Computer Science & Engineering for giving me valuable suggestions and critical inputs in the preparation of this report. I am also thankful to Dr. Niraj Upadhayaya, Head of Department of Computer Science & Engineering for encouragement.

My friends in my class have always been helpful and I am grateful to them for patiently listening to my presentations on my work related to the Project.

Lovely Yeswanth Panchumarthi, Sriya Padmanabhuni, Lavanya Parchuri,
Yogeshvar Reddy Kallam
(Reg. No. AP20110010299, AP20110010274, AP20110010233,
AP20110010145)
B. Tech.
Department of Computer Science & Engineering
SRM University-AP

ABSTRACT

Neural Information Retrieval (NIR) has revolutionized the field of information retrieval, offering new methodologies and techniques to enhance search capabilities. This abstract explores four key papers that contribute to the advancement of NIR. The first paper conducts a comprehensive literature review on word embeddings, highlighting their pivotal role in capturing semantic relationships and improving search accuracy. The second paper introduces a modified cross encoder for two-stage passage ranking, combining traditional ranking methods with modern transformer-based architectures to achieve superior retrieval performance. The third paper proposes a novel approach using asymmetric divergence for query expansion, aiming to enhance personalized search systems by expanding queries with semantically related terms. Finally, the fourth paper focuses on text classification, showcasing the significance of categorizing textual data accurately to streamline information retrieval processes. Collectively, these papers demonstrate the diverse applications of neural networks in NIR and underline the continuous evolution of information retrieval techniques.

CONTENTS

LIST OF TABLES

LIST OF FIGURES

Chapter 1

INTRODUCTION

In recent years, the integration of neural network methodologies into Information Retrieval (IR) systems has sparked a significant revolution, particularly in the subfield of Neural Information Retrieval (NIR). This integration has opened up new avenues for enhancing search capabilities and understanding user queries, leading to profound advancements in the field. Neural networks have been instrumental in advancing areas of NIR, including word embedding, passage ranking, query expansion, and text classification. These advancements have led to more efficient and effective IR systems, capable of providing users with more relevant and accurate search results. In this paper, we focus on four key topics in Neural Information Retrieval: Word Embedding, Passage Ranking, Asymmetric Divergence from Randomness in Query Expansion, and Text Classification.

1.1 WORD EMBEDDING

Word embeddings have transformed the field of NIR by enabling us to comprehend and handle language data in a completely new way. This study explores several embedding models that have played a key role in the development of the subject. The breakthrough of word embedding came in the 2000s, when several neural network-based models were proposed to learn word vectors from large-scale text corpora. These models can be broadly classified into two categories: count-based models and predictive

models. Count-based models, such as GloVe [113] and Hellinger PCA, learn word vectors by factorizing a global co-occurrence matrix, which encodes the statistics of word occurrences in the whole corpus. Predictive models, such as word2vec [108, 109] and fastText [144], learn word vectors by optimizing a local objective function, which predicts the surrounding words of a given word in a sliding window. Both types of models have shown to produce high-quality word vectors that can capture various linguistic regularities and semantic relations. However, they also have some drawbacks, such as the inability to handle polysemy, the ignorance of word order and context, and the reliance on large amounts of unlabeled data.

To address these issues, several extensions and variations of word embedding models have been proposed in recent years. One direction is to incorporate external knowledge sources, such as dictionaries, ontologies, and knowledge graphs, into word embedding models, to enhance the semantic richness and diversity of word vectors. For example, ERNIE [172] and KEPLER leverage entity and relation information from knowledge graphs to improve word embedding. Another direction is to develop contextualized word embedding models, which can generate dynamic word vectors that adapt to different contexts and senses. For example, ELMo [157] and BERT use deep bidirectional language models to encode the contextual information of words. Pre-trained Language Models (PLMs) [157, 175, 176] are NLMs that are pre-trained on large-scale text corpora using self-supervised learning objectives, such as masked language modeling (MLM) or next sentence prediction (NSP). PLMs can generate contextualized word embeddings, which capture the dynamic meaning of words depending on their context. A third direction is to extend word embedding to other levels of linguistic units, such as characters, sentences, documents, and graphs. For

example, Char2Vec and BytePair learn character-level embeddings to handle rare and out-of-vocabulary words. Doc2Vec [96] and Sent2Vec [95] learn sentence and document embeddings to represent longer texts. Graph2Vec [153] and GraphSAGE learn graph embeddings to capture the structural and semantic information of graphs.

Word embedding models have been applied to various NIR tasks, such as query expansion [133], document representation [117], similarity scoring, and semantic compositionality. Word embedding models have shown to improve the effectiveness and efficiency of NIR systems, by providing more expressive and compact representations of words and texts. However, there are also some challenges and open questions, such as how to evaluate word embedding models, how to choose the optimal dimensionality and granularity of word vectors, how to integrate word embedding models with other IR models and features, and how to deal with the dynamic and evolving nature of language and information.

1.2 PASSAGE RANKING

Passage ranking [1] [2] [3] is a fundamental task in information retrieval that aims to identify and rank relevant passages within documents in response to user queries. Unlike document ranking [4] [5], which focuses on entire documents, passage ranking operates at a finer granularity, seeking to extract and prioritize specific passages that best address the user's information needs [6]. This task is crucial in scenarios where users seek specific information or answers within large documents, such as in web search [8], question answering [9], and document summarization [10].

In the early days of information retrieval, algorithms like TF-IDF [48], BM25 [47], Boolean Retrieval [11], Vector Space Model [12], and Okapi BM25

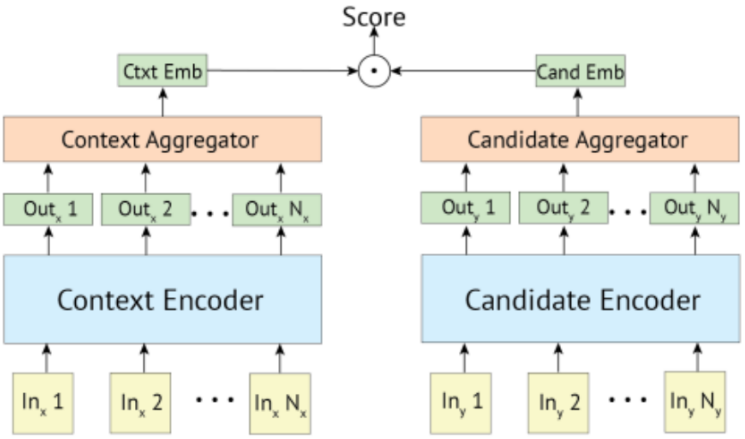

Figure 1.1: BiEncoder Architecture

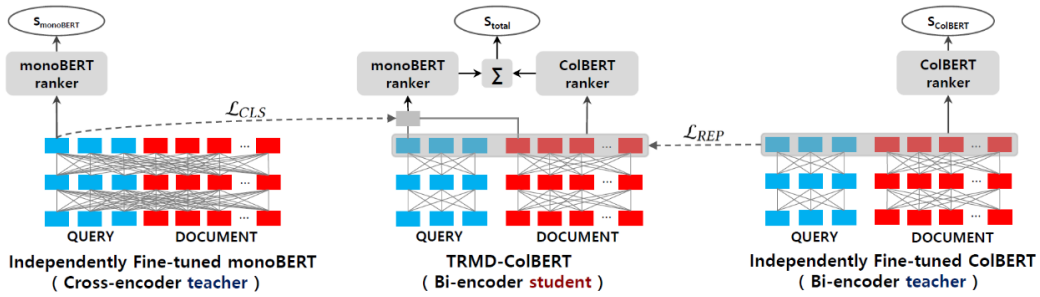

Figure 1.2: Encoder Model Architectures

[13] were frequently employed for passage ranking. These methods, though straightforward, played a foundational role by emphasizing term weighting, document similarity, and term matching to rank passages based on user queries. However, they are considered less effective for more complex information retrieval tasks due to their inability to capture semantic relationships and contextual understanding.

Sparse [16] and dense [17] models complement each other in passage re-ranking. These models optimize computational resources while maximizing retrieval effectiveness as shown in figure 1.1. Sparse models i.e SpaDe [18] efficiently identify and prioritize relevant passages, enhancing precision and offering insights into ranking criteria interpretation . Conversely,

dense models i.e Simlm [19] capture complex semantic relationships, improving accuracy by utilizing dense, continuous representations. This nuanced understanding enables precise rankings, further enhanced through fine-tuning on specific datasets, making dense models indispensable for improving search result quality in information retrieval systems.

In the context of passage re-ranking, hybrid cross models like RocketQAv2 [20], offer a unique advantage by integrating sparse and dense model components. These models efficiently capture key information using sparse features while also incorporating dense representations for a more comprehensive understanding of context. This dual approach enables cross-models to strike a balance between efficiency and effectiveness in identifying and prioritizing relevant passages. The advent of neural networks has sparked a revolution across various domains, with Information Retrieval (IR) being no exception.

1.3 PERSONALIZED SEARCH

Personalized search [37] [38], [39] evaluation is a fundamental aspect of Information Retrieval (IR) research, aiming to customize search results to meet the unique preferences and needs of individual users. The evaluation of personalized search [37] systems presents several challenges, particularly in capturing the intricate semantic relationships between terms and the context of the query. While traditional retrieval models have made strides in addressing some of these challenges, they often fall short in fully grasping the complex interplay between terms and the query context [40]. Personalized search systems play a crucial role in modern information retrieval, as they aim to deliver tailored search results that are more relevant to users' specific needs.

The concept of personalization in the field of Information Retrieval has been extensively researched by both academic scholars and industry professionals. This research has spanned various aspects, including the development of models and the evaluation of their performance [41]. It is widely recognized that the traditional Cranfield evaluation paradigm is not ideally suited for Personalized Search. This is because Personalized Search aims to deliver search results that are customized to the unique needs and preferences of individual users, which is a deviation from the one-size-fits-all approach that the Cranfield paradigm is based on [42]. Unfortunately, the text collections that are commonly shared to evaluate search algorithms often lack the necessary information for evaluating personalization [43]. Specifically, they do not provide information about individual users and their unique preferences. In this paper, we present an alternative approach to enhancing personalized search systems by implementing Query Expansion (QE) [44] using Kullback-Leibler Divergence (KLD). Specifically, we focus on Gianni Amati's (GA) [45] method is described in detail in section 3, which augments retrieval model performance by expanding the query with terms that are semantically related to the original query, aiming to improve the alignment between the query and the documents retrieved.

The approach by GA is noteworthy for its emphasis on terms specific to feedback documents, which guides the retrieval process towards informative documents for the user's query. This method, rooted in information theory, provides a robust mathematical framework for measuring term informativeness. In our implementation using PyTerrier [46], we applied a class that implements the KL query expansion model from the Divergence from Randomness Framework. Our implementation, along with detailed documentation and associated run scripts available in our GitHub, ensures

experimental results that are both repeatable and replicable.

1.4 TEXT CLASSIFICATION

Text classification [69, 70, 71, 72] is a cornerstone technique in the field of information retrieval, serving as a vital tool for organizing and categorizing textual data to streamline the process of information retrieval. By automatically assigning predefined categories or labels to text documents, text classification significantly enhances the efficiency of search operations, allowing users to swiftly locate relevant information. Text relies on a variety of algorithms, chosen based on task complexity and text data characteristics. Common algorithms include Decision Trees, Gradient Boosting Machines (GBM) [73], Random Forests, Convolutional Neural Networks (CNNs) [74], and Recurrent Neural Networks (RNNs) [75]. Pre-trained models such as BERT [76] and GPT [77], and their variants, are popular for their ability to leverage large-scale pre-training on text corpora, improving performance on downstream tasks. Non-pretrained models, like Simple Logistic Regression and Multinomial Naive Bayes, are trained from scratch and used when pre-trained models are not available or suitable.

In the realm of information retrieval, non-pre-trained models have proven their mettle by effectively capturing textual patterns and semantics without extensive pre-training. Notable architectures include Text_RCNN, which combines Recurrent Neural Networks (RNNs) and Convolutional Neural Networks (CNNs) to model sequential and local patterns respectively. Fast-Text_DPCNN [78] leverages the efficient fastText library and Deep Pyramid Convolutional Neural Networks (DPCNNs) for feature extraction. The FastText model [79] itself, known for its lightweight and efficient nature, has been widely adopted for text classification tasks. Ad-

ditionally, Text_RNN [80] employs RNNs to capture sequential dependencies, while Text_RCNN_Att [81] incorporates attention mechanisms to the Text_RCNN architecture, allowing it to focus on the most relevant parts of the input text. These non-pre-trained models have demonstrated their efficacy in various information retrieval tasks, such as document categorization, spam filtering, and sentiment analysis, by effectively capturing textual patterns and semantics without the need for extensive pre-training.

Pre-trained models have revolutionized text classification in information retrieval. Unlike non-pre-trained models, which learn from scratch, pre-trained models leverage knowledge from large corpora for better generalization and higher accuracy. BERT (Bidirectional Encoder Representations from Transformers) [82] [83] [84] is a standout model in this regard, capturing bidirectional context and serving as the basis for various architectural variations. BERT_CNN [85] incorporates Convolutional Neural Networks (CNNs) for local pattern capture, while BERT_DPCNN uses Deep Pyramid CNNs (DPCNNs) [86] for enhanced feature extraction. BERT_RNN [87] integrates Recurrent Neural Networks (RNNs) for sequential dependency modeling. The Transformer [88] architecture, introduced with BERT, has gained traction for its self-attention mechanisms, effectively modeling long-range dependencies. These pre-trained models, including BERT and its variants, have significantly improved performance in sentiment analysis, topic categorization, and spam detection, transforming the field of information retrieval.

Chapter 2

LITERATURE SURVEY

2.1 LITERATURE REVIEW OF WORD EMBEDDING

The work aims to survey the current state-of-the-art of word embedding methods for Information Retrieval (IR), focusing on the different types of word embedding methods, their applications, and their evaluation protocols. We present word embedding models into two classes: embedding models for categorical data and embedding models for non-categorical data. We also discuss the challenges and opportunities of contextualized, sentence, document, and graph embedding methods. We then survey the main applications of word embedding in NIR, such as query expansion, document representation, similarity scoring, and semantic compositionality. We also present the evaluation protocols, benchmark datasets, and complexity analysis of word embedding methods.

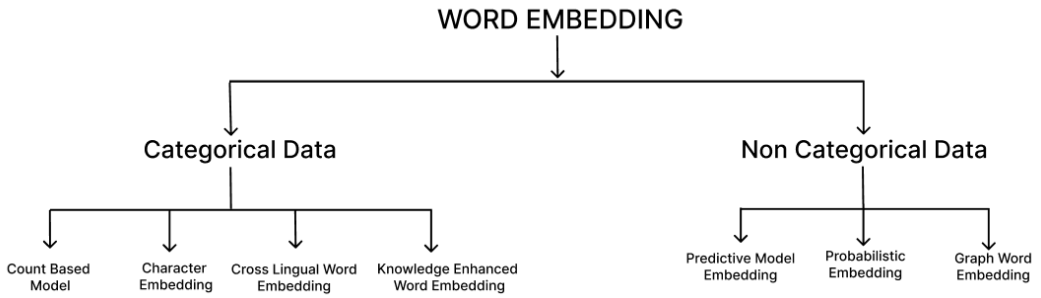

Figure 2.1: Categories of Word Embedding

2.2 EMBEDDING MODELS FOR CATEGORICAL DATA

Categorical data embedding models are techniques that strive to map categorical data into a continuous vector space, ensuring that categories with similar characteristics are represented by similar vectors in the embedding space. This category has seen numerous research initiatives aimed at developing a variety of models to encapsulate as much of the categorical data's semantic information as possible.

We categorize the embedding models used for learning category representations into four types: (i) Character Embedding, which considers local character proximity, including first-order, second-order, and high-order proximity, (ii) Cross-lingual Word Embedding, which accounts for structural role proximity and intra-language proximity, (iii) Knowledge Enhanced Word Embedding, which encapsulates global language properties like the scale-free property or small world property, and (iv) Count-Based Model, which is predicated on the frequency of category occurrences.

2.2.1 Count-Based Embedding

In the realm of word embeddings, a significant approach is the utilization of Count-Based Models. These models operate on the principle of the distributional hypothesis, which postulates that words appearing in similar contexts are likely to bear similar meanings. One of the key techniques employed in Count-Based Models is the computation of Pointwise Mutual Information (PMI). PMI quantifies the likelihood of the co-occurrence of two words compared to their probabilities of occurrence. By weighting the co-occurrence matrix with PMI, the impact of high-frequency words can be mitigated, thereby enhancing the quality of the word embeddings.

To manage the high-dimensionality of the co-occurrence matrix, Di-

mensionality Reduction techniques are employed. These techniques aim to preserve the essential information while reducing the number of dimensions. A common method used for this purpose is Singular Value Decomposition (SVD), which breaks down a matrix into three smaller matrices. An alternative to SVD is Random Indexing [98], a method that incrementally constructs word vectors. Each word is assigned a sparse random vector, and the vectors of the context words are added to the target word vector. While Random Indexing method offers advantages in terms of speed and scalability, it may lose some information due to collisions. In the context of IR, these Count-Based Models play a crucial role in capturing semantic and syntactic relationships between words, thereby improving the effectiveness of IR systems. Some of the models of Count-Based model category are discussed below:

2.2.1.(i) GloVe: Global Vectors for Word Representation

GloVe [113] is an unsupervised learning algorithm that obtains vector representations for words by exploiting the global word-word co-occurrence statistics from a large corpus of text . The key idea behind GloVe is that the ratios of co-occurrence probabilities can encode some form of meaning that can be captured by vector differences. It proposes a log-bilinear model that can efficiently capture the global co-occurrence statistics in a word vector space. The model is defined by the following equation:

$$w_i^T \tilde{w}_k + b_i + \tilde{b}_k = \log X_{ik} \tag{2.1}$$

where w_i and $\tilde{w}_k$ are the word vectors for words i and k, b_i and $\tilde{b}_k$ are the corresponding biases, and X_{ik} is the number of times word k occurs in the context of word i.

2.2.1.(ii) RAND-WALK

This paper's method [122] is a latent variable model approach to word embeddings, which aims to capture the semantic and syntactic relationships between words based on their co-occurrence statistics in a corpus. In the model, the generation of a corpus is driven by a random walk over a latent discourse space, where each word has a latent vector that represents its correlation with the discourse vector. The probability of a word given the discourse vector is proportional to the exponential of their inner product. The random walk is assumed to have a uniform stationary distribution over the unit sphere, and the word vectors are assumed to be isotropic, meaning that they have no preferred direction in space. The equation for co-occurrence probability model is:

$$\log p(w, w') = \frac{\|v_w + v_{w'}\|^2}{4d} - 2\log Z \pm \epsilon \tag{2.2}$$

where $p(w, w')$ is the probability that words w and w' occur next to each other in a window of size $q = 2$, Z is the partition function, and ϵ is a small error term. This equation represents the log probability of co-occurrence for words w and w' based on the sum of their word vectors in the discourse space. The model considers the Euclidean norm of the sum of the word vectors, the dimension of the space, and a partition function that helps normalize the probability distribution. The small error term ε introduces some flexibility or noise into the mode.

2.2.1.(iii) Hellinger PCA

In the paper [214], model use the word co-occurrence statistics to acquire representations of word meanings. The author proposes to perform

a Hellinger PCA of the word co-occurrence probability matrix, which preserves the Hellinger distance between word distributions. The Hellinger distance between two discrete probability distributions $P = (p_1, \ldots, p_k)$ and $Q = (q_1, \ldots, q_k)$ is defined as:

$$H(P,Q) = \frac{1}{\sqrt{2}} \sqrt{\sum_{i=1}^{k} (\sqrt{p_i} - \sqrt{q_i})^2} \tag{2.3}$$

2.2.2 Character embeddings

In another category of word embedding as Character based word embedding, simply we call it as Character Embedding in this report. Character embeddings have emerged as a powerful technique in the field of IR, particularly in the context of word embedding models [131]. Unlike traditional word embeddings that treat words as atomic units, character embeddings represent words as sequences of characters. This approach effectively handle out-of-vocabulary words which is a common challenge in IR by capturing subword information [141]. One notable model that leverages character embeddings is the Character-Aware Word Embeddings model [131]. This model combines character embeddings with word embeddings, using a convolutional neural network (CNN) to encode character sequences into word vectors. The integration of character-level information into word embeddings has been shown to improve performance on various Natural Language Processing (NLP) tasks, such as part-of-speech tagging and named entity recognition [129]. Some of the models of Character model category are discussed below:

Many of the earlier works [...] proposed various character-based word embedding models and these models were employed for various application tasks [...] related to NLP and IR. The following sub sections will discuss the

theoretical details of these models.

2.2.2.(i) Subword Skipgram Model

To capture the internal structure of a word, this method learn the word embeddings by incorporating subword information [144]. So that, this model represent a word as the sum of the vector representations of its $n-grams$. To learn these representations using the skipgram model, it aims to maximize the below log-likelihood:

$$\sum_{i}^{N} \sum_{c \in C_i} \log P(x_c|x_i), \tag{2.4}$$

where N is the total unique terms in vocabulary, x_c is a context word to the word x_i and the probability function is:

$$P(x_c|x_i) = \frac{e^{s(x_i,x_c)}}{\sum_{j=1}^{N} e^{s(x_i,x_j)}}, \tag{2.5}$$

where s is a score function defined as $s(x_i, x_c) = \sum_{g \in G_{x_i}} v_g^T \cdot v_c$. Here v_g is the vector representation of n-gram g and v_c is the vector representation of word x_c. While working with morphologically rich languages, most of the earlier works [121, 159] for various applications related to NLP employed this model to handle *out-of-vocabulary* problems.

2.2.2.(ii) charn-MS-vec

Inspired by advancements in word embedding construction [135], for exploiting character information into various NLP related applications, the method [187] employe the Multi-dimensional Self-attention (MS) on character n-gram embeddings to encode them into a word embedding. The

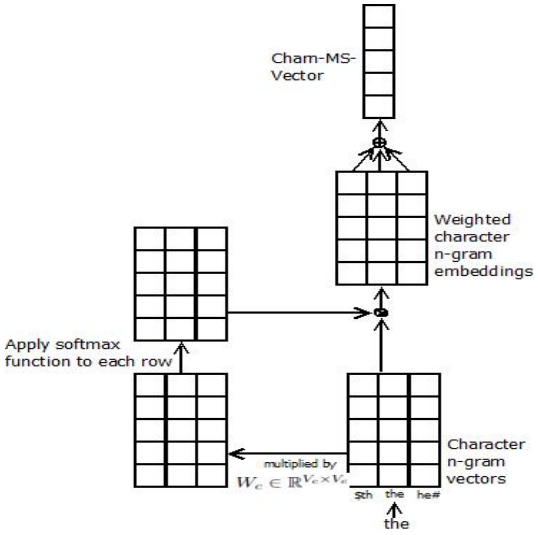

Figure 2.2: The process of Char$n$-MD-Vec

equation for charn-MS-vec is:

$$c_t = \sum_{i=1}^{I} g_i \odot s_i \tag{2.6}$$

where c_t is the charn-MS-vec for the word at time-step t, I is the number of character n-grams extracted from the word, g_i is the weight vector computed by applying softmax to each row of $W_c S^T$, s_i is the character n-gram embedding, $\odot$ is the element-wise product, and W_c is a weight matrix. The process of this method is given in the following fig 2.2.

This enhance the representation of less common words by capturing internal structures through character n-gram embeddings [121]. The adoption of multi-dimensional self-attention mitigates information loss during aggrigation [159].

2.2.2.(iii) Affect Character and Word Embeddings (ACWE)

This approach, ACWE [189], provides a morphological and semantic comprehensive representation of each word by combining the standard

word embedding (WE) from Word2Vec and character embedding (CE) from FastText approaches. This method addresses three key scenarios: (i) direct concatenation if the word exists in both CE and WE, (ii) if the word is absent in WE then substitution with the most similar word from CE, and (iii) If word not avialable in both CE and WE, then utilize a vector of zeros. This innovative approach showcases the adaptability and robustness in handling diverse linguistic challenges. Making it a noteworthy contribution to the affective analysis of code-mixed social media text such as Arabic-English, Bengali-English, Hindi-English, and Telugu-English. The ACWE method can be formally defined as follows:

$$\text{ACWE}(x_i) = \begin{cases} \text{CE}(x_i) \oplus \text{WE}(x_i), & \text{if } x_i \in (\text{CE}_{|V|}, \text{WE}_{|V|}) \\ \text{CE}(x_i) \oplus \text{WE}(\text{mostSimilar}(x_i)), & \text{if } x_i \notin (\text{WE}_{|V|}) \\ \text{zeros of}(\text{CE} + \text{WE}) \text{ dimensions}, & \text{otherwise} \end{cases} \tag{2.7}$$

where x_i is the i-th word in a tweet, $\text{CE}(x_i)$ and $\text{WE}(x_i)$ are the character and word embeddings of x_i, respectively, $\oplus$ is the concatenation operator, and $\text{mostSimilar}(x_i)$ is the function that returns the most similar word to x_i in CE. This method is evaluated and presented the results with the state-of-the-art Arabic pre-trained word embeddings, such as AraVec [148], AraELECTRA [190], and AraBERT [190]. Also evaluated on state-of-the-art Bengali-English, Hindi-English and Telugu-English code-mixed twitter data. This method also compared with the top-performing systems that participated in the SemEval-2018 Task 1: Affect in Tweets [161].

2.2.3 Knowledge-enhanced embeddings

Knowledge-enhanced word embeddings are a class of word embedding models that leverage external knowledge sources to enrich the semantic representation of words. These models aim to address some of the limitations of conventional word embedding models, such as the inability to handle polysemy, the ignorance of word order and context, and the reliance on large amounts of unlabeled data [178]. Knowledge-enhanced word embeddings can be categorized according to the type of knowledge source, the method of knowledge injection, and the level of granularity of the word embeddings [166].

The type of knowledge source refers to the form and content of the external knowledge that is used to enhance the word embeddings. Some common types of knowledge sources are semantic graphs, semantic lexicons, and semantic constraints. Semantic graphs are graph structures that represent semantic relations between concepts, such as WordNet, ConceptNet, or knowledge bases [132]. Semantic lexicons are collections of words or phrases that are grouped by semantic similarity or relatedness, such as WordNet synsets, PPDB paraphrases [110], or Babel domains [134]. Semantic constraints are pairs of words or phrases that are annotated with a semantic relation or a similarity score, such as word similarity datasets, word analogy datasets, or semantic textual similarity datasets [94, 108].

The method of knowledge injection refers to the way the external knowledge is incorporated into the word embedding learning process. Some common methods of knowledge injection are joint learning, post-processing, and pre-training. Joint learning is a method that learns word embeddings and knowledge embeddings simultaneously from scratch, using a joint objective function that combines corpus-based and knowledge-based signals.

Post-processing is a method that modifies pre-trained word embeddings to align them with the knowledge embeddings, using linear transformations, graph propagation, or retrofitting methods [124, 143]. Pre-training is a method that learns knowledge embeddings from a knowledge source and uses them as initialization for word embeddings, which are then fine-tuned on a corpus [160, 76, 171].

The level of granularity of the word embeddings refers to the size and scope of the linguistic units that are represented by the word embeddings. Some common levels of granularity are word, subword, and phrase. Word-level embeddings are the most common and represent each word by a single vector [109, 113]. Subword-level embeddings are an extension of word-level embeddings that incorporate subword information, such as characters, n-grams, or morphemes, into word embeddings [141, 131, 136]. Phrase-level embeddings are another extension of word-level embeddings that represent multi-word expressions, such as idioms, collocations, or named entities, by a single vector [128, 156].

2.2.3.(i) Logic Embeddings (Log-Emb)

A method that learns low-dimensional embeddings of entities and first-order logic (FOL) queries that can simulate the behavior of FOL [195]. The key theory behind the model is that FOL queries can be represented as linear operators on the embedding space, and the embeddings of entities and queries can be learned by minimizing the reconstruction error between the original and the reconstructed queries. The equation of the model is

$$\mathcal{L}(\theta) = \sum_{q \in \mathcal{Q}} \sum_{x \in \mathcal{X}} \left\| q(x) - \hat{q}(x) \right\|^2 \tag{2.8}$$

where θ is the model parameters, $\mathcal{Q}$ is the set of FOL queries, $\mathcal{X}$ is the

set of entities, $q(x)$ is the truth value of query q applied to entity x, and $\hat{q}(x)$ is the reconstructed truth value computed by applying the linear operator $\hat{q}$ to the embedding of entity x. The model learns the embeddings of entities and queries by minimizing this loss function using stochastic gradient descent.

2.2.3.(ii) AWE-CM Vectors

Augmenting Word Embeddings with a Clinical Metathesaurus [146] involves enhancing word embeddings with domain knowledge from the Unified Medical Language System (UMLS) metathesaurus. The model is related to Knowledge Enhanced Word Embeddings as it incorporates expert domain knowledge into the embeddings. The paper uses an extension of word2vec, a popular tool for learning word embeddings from text, that allows for arbitrary features as contexts. This means that the model can learn from both the words that appear near a word in a sentence, and the concepts that are associated with a word in the metathesaurus.

The paper trains word embeddings on a large corpus of clinical notes from the MIMIC-III database, using both the word-based and the concept-based contexts. The paper calls the resulting embeddings AWE-CM, which stands for Augmenting Word Embeddings with a Clinical Metathesaurus. The paper shows that the AWE-CM embeddings outperform the text-only embeddings on all three sets, and achieve the best correlation with the physician judgments.

2.2.3.(iii) Constrained CBOW

A new methodology [218] by incorporating prior knowledge from the medical domain to improve medical information retrieval (MIR). The key theory behind the model is based on the principle that related words,

such as synonyms, should have similar embeddings. The model introduced is related to Knowledge Enhanced Word Embedding. It leverages existing knowledge in the form of word relations to constrain the embeddings, ensuring that semantically related words are closer in the embedding space. The equation for the modified loss function in the constrained word embedding model is:

$$L = \sum_{t=1}^{T} \left[\frac{1}{|R_t|} \sum_{(w_t, w_s) \in R} (\log p(w_t \mid w_{t-k}) - \log p(w_s \mid w_{t-k}))^2 \right] \tag{2.9}$$

where T is the total number of words in the corpus, R_t is the number of words related to w_t in the resource, and $p(w_t \mid w_{t-k})$ is the probability of a word given its context. This equation represents a tighter regularization in the online method, where the original CBOW cost function is combined with the requirement that if a word can be well generated from a given context, its related word should also be well generated from the same context.

2.2.3.(iv) Seeded Biterm Topic Model (SeedBTM)

This model [127] integrates topic model information, specifically from Latent Dirichlet Allocation (LDA), into the Skip-gram model to capture precise topic-based word relationships for text classification tasks. This model integrates Latent Dirichlet Allocation (LDA) with the entire text corpus to discern topic-based semantic relationships among words. This inclusion of LDA allows the model to capture nuanced contextual information within the dataset. It introduces a pioneering objective function tailored for the Skip-gram model. This novel function directs the model towards predicting words that share similar topic-based semantic information.

The resulting framework, termed the Topic-based Skip-gram model,

establishes a connection with the broader domain of Knowledge Enhanced Word Embedding. This is evident in the model's augmentation of the Skip-gram approach with knowledge derived from topic models like LDA, representing a form of external semantic knowledge. In essence, the integration of LDA and the novel objective function enhances the Skip-gram model by incorporating external topic-related information, thus contributing to a more comprehensive and contextually informed word embedding paradigm. The probability of a word given its context in the Topic-based Skip-gram is defined using the softmax function:

$$p(w_O|w_I) = \frac{\exp({v'_{w_O}}^T v_{w_I})}{\sum_{w=1}^{W} \exp({v'_w}^T v_{w_I})} \tag{2.10}$$

where w_O is the output word, w_I is the input word, v'_w and v_w are the 'output' and 'input' vector representations of the word w, and W is the number of words in the vocabulary.

2.2.3.(v) Knowledge Enhanced Contextual Word Representations (KnowBert)

This [183] model integrates knowledge bases (KBs) into large pre-trained language models like BERT to enhance their representations with structured, human-curated knowledge. KnowBert employs a Knowledge Attention and Recontextualization (KAR) mechanism. It models entity spans in the input text and uses an entity linker to retrieve relevant entity embeddings from a KB, forming knowledge-enhanced entity-span representations. These are then recontextualized with word-to-entity attention, allowing interactions between word representations and all entity spans in the context. The model uses a transformer block with a multi-headed self-attention layer followed by a position-wise multilayer perceptron (MLP).

The equation for the transformer block is

$$TransformerBlock(H_{i-1}) = MLP(MultiHeadAttn(H_{i-1}, H_{i-1}, H_{i-1})) \quad (2.11)$$

where H_{i-1} represents the contextual representations from the previous layer.

2.2.3.(vi) Knowledge-based Inference Model (KIM)

A methodology for enhancing neural natural language inference (NLI) models with external knowledge [163]. The model involves incorporating lexical-level semantic knowledge into major components of neural networks for NLI. The equation for the model's co-attention mechanism is given by:

$$e_{ij} = (a_i^s)^T b_j^s + F(r_{ij}) \quad (2.12)$$

where e_{ij} is the co-attention between the i-th word in the premise and the j-th word in the hypothesis, a_i^s and b_j^s are the context-dependent hidden states, and r_{ij} is the relation feature derived from external knowledge. The function F can be a non-linear or linear function applied to r_{ij}.

2.2.3.(vii) Knowledge-Enriched Ensemble

A method [198], is designed to amalgamate insights from knowledge graphs and pre-trained word embeddings. While the approach lacks a specific nomenclature, it revolves around the utilization of an attention network as its central mechanism. This attention network plays a pivotal role in seamlessly incorporating semantic information extracted from a lexical knowledge graph into existing pre-trained word embeddings.

The model, distinguished by its integration of external semantic sources

from knowledge graphs, is positioned within the realm of Knowledge Enhanced Word Embedding. This infusion of semantic knowledge proves especially advantageous in tasks demanding an in-depth comprehension of word meanings and intricate contextual relationships.

2.2.3.(viii) Knowledge Enhanced Contextual Word Representation (KECWR)

The paper [194] delves into the augmentation of word embeddings through the integration of additional morphological, syntactic, semantic, and domain knowledge. The primary objective is to enhance the quality of embeddings for diverse Natural Language Processing (NLP) tasks. It acknowledges that conventional word embeddings, while adept at capturing semantic relatedness, may fall short in effectively encoding other essential types of information crucial for a range of NLP applications.

No specific equation is given but it discusses methodologies such as joint optimization, post-processing, and solo or coupled embedding techniques for integrating knowledge into embeddings. Categorized under Knowledge Enhanced Word Embedding, the model aims to enrich both static word embeddings like Word2Vec and dynamic/contextual embeddings like BERT. External knowledge sources, including but not limited to Wikipedia and WordNet, are leveraged to augment the embeddings. It underscores the significance of incorporating diverse types of knowledge, emphasizing that such an approach can markedly enhance the performance of word embeddings across a spectrum of NLP tasks.

2.2.3.(ix) knowledge-powered deep learning

The paper [118] discusses for word embedding that leverages morphological, syntactic, and semantic knowledge to enhance the quality of

word embeddings. The model involves in incorporating this knowledge into the deep learning process to define new bases for word representation, provide additional input information, and serve as auxiliary supervision.

The study leverages morphological knowledge by incorporating root/affix and syllable information to establish new bases for word representation. This approach serves the dual purpose of reducing vocabulary size while preserving word semantics. Additionally, the integration of syntactic and semantic knowledge acts as supplementary input and auxiliary supervision, contributing to the enhancement of word embeddings' effectiveness in capturing intricate word similarities and relationships. The Continuous Bag-of-Words Model (CBOW) forms the foundational model for this investigation, serving as the base upon which the proposed methods are applied to achieve improved performance in various Natural Language Processing (NLP) tasks. The synergy of morphological insights and syntactic/semantic knowledge within the CBOW framework signifies a comprehensive strategy aimed at refining word embeddings for enhanced capabilities in NLP applications.

Equation of Main Objective (LM)

$$LM = \frac{1}{X}\sum_{x=1}^{X} \log p(w_x|W_{d_x}) \tag{2.13}$$

Equation of Objective for Entity Knowledge (LE)

$$LE = \frac{1}{X}\sum_{x=1}^{X}\sum_{k=1}^{K} 1(w_x \in e_k) \log p(e_k|W_{d_x}) \tag{2.14}$$

Equation of Objective for Relation Knowledge (LR)

$$LR = \frac{1}{X}\sum_{x=1}^{X}\sum_{r=1}^{R}\lambda_r\sum_{n=1}^{N} r(w_x, w_n)\log p(w_n|W_{d_x} \tag{2.15}$$

These equations are part of a multi-task learning framework where the main objective is to predict the center word given its context, and the auxiliary objectives incorporate additional knowledge to improve the quality of the word embeddings. The paper provides a detailed empirical study on the effectiveness of this knowledge-powered deep learning approach.

2.2.3.(x) ERNIE: Enhanced Language Representation with Informative Entities

The paper [182] discusses a new methodology for language representation. It aims to enhance language models by incorporating knowledge graphs (KGs) to provide structured knowledge facts for better language understanding. At the core of ERNIE lies a foundational theory centered on the holistic integration of lexical, syntactic, and knowledge information extracted from both textual corpora and Knowledge Graphs (KGs). This integration is achieved through a multi-faceted approach.

Textual Encoder

$$\{w_1, \ldots, w_n\} = \textit{T-Encoder}(\{w_1, \ldots, w_n\}), \tag{2.16}$$

where $\textit{T-Encoder}(\cdot)$ is a multi-layer bidirectional Transformer encoder that computes lexical and syntactic features from token sequences.

Knowledgeable Encoder

$$\{w_1^o, \ldots, w_n^o\}, \{e_1^o, \ldots, e_n^o\} = \textit{K-Encoder}(\{w_1, \ldots, w_n\}, \{e_1, \ldots, e_m\}). \tag{2.17}$$

This encoder fuses the textual information with knowledge embeddings from KGs to produce final output embeddings for tokens and entities.

Firstly, ERNIE aligns named entity mentions in text with corresponding entities in KGs, employing knowledge embedding algorithms such as TransE to encode the graph structure of the KGs. This aligns the model's understanding of entities in the text with the structured knowledge available in KGs. Secondly, ERNIE adopts a comprehensive set of pre-training objectives, including a masked language model and next-sentence prediction. An innovative addition involves masking some named entity alignments and tasking the model with selecting appropriate entities from KGs to complete these alignments. Finally, the model integrates this wealth of information through a knowledgeable encoder (K-Encoder), harmonizing textual and knowledge features. The output of this fusion process yields final embeddings tailored for specific tasks, showcasing ERNIE's prowess in leveraging both textual and knowledge-based information for enhanced performance in various natural language understanding tasks.

Denoising Entity Auto-encoder (dEA)

$$p(e_j|w_i) = \frac{\exp(\text{linear}(w_i^o) \cdot e_j)}{\sum_{k=1}^{m} \exp(\text{linear}(w_i^o) \cdot e_k)}, \tag{2.18}$$

where $p(e_j|w_i)$ is the probability distribution of aligned entities for a token w_i, and the model is trained to predict entities based on aligned tokens. ERNIE belongs to the category of Knowledge Enhanced Word Embedding models, as it explicitly incorporates external knowledge from KGs into the language representation model.

2.2.4 Cross-lingual Word Embedding

Cross-lingual Word Embedding is an important area of research in NLP that aims to capture semantic relationships between words in different languages. This technique enables the representation of words from multiple languages in a shared vector space, allowing for cross-lingual comparisons and IR. One of the basic models introduced in this field is the Bilingual Word Embeddings (BiWE) model [139]. This model uses parallel bilingual corpora to learn word embeddings that are aligned across languages. This model has been successful in capturing cross-lingual similarities and has been applied in various NLP tasks such as sentiment analysis [184] and machine translation [185].

There are few other advanced model for crosslingual word representing. One such model is the Cross-lingual Word Embedding through Model Alignment (CLWE-MA) [105]. This model leverages monolingual word embeddings and a bilingual dictionary to align the word embeddings of different languages. Another model, named as the Cross-lingual Word Embedding with Adversarial Training (CLWE-AT) [145], uses an adversarial training approach to project words from different languages into a shared embedding space.

These models have found applications in various domains. For instance, cross-lingual word embeddings have been used for cross-lingual information retrieval [105], cross-lingual document classification [139], and cross-lingual sentiment analysis [184]. Recent methodologies in this field include the use of graph-based approaches for learning cross-lingual embeddings [158] and the integration of external knowledge sources like Wikipedia to enhance the quality of cross-lingual word embeddings [174]. Details of each of these proposed models are explained in the following subsections.

2.2.4.(i) Word Embeddings for REST Services Discoverability (WERSD)

A new methodology for semantic annotation of REST services using word embeddings. The key theory behind the model is that word embeddings can capture the semantic similarity between words and phrases, and thus can be used to annotate the inputs and outputs of REST services with concepts from a domain ontology [150]. The equation of the model is:

$$sim(s,c) = \frac{1}{n}\sum_{i=1}^{n}\max_{j=1}^{m} cos(w_i, c_j) \tag{2.19}$$

where s is a service description, c is a concept from the ontology, n is the number of words in s, m is the number of words in c, w_i is the i-th word in s, c_j is the j-th word in c, and cos is the cosine similarity between word embeddings. This paper uses cross-lingual word embeddings to enable multilingual annotation of REST services, meaning that services can be annotated with concepts from different languages. A projection-based approach is used to obtain cross-lingual word embeddings, which consists of learning a linear transformation that maps word embeddings from one language to another. This is the following equation to learn the transformation matrix W:

$$W = \arg\min_{W} \sum_{(x,y)\in D} ||Wx - y||^2 \tag{2.20}$$

where D is a set of word pairs (x, y) that are translations of each other, and x and y are word embeddings from different languages.

2.2.4.(ii) Cross-Language Latent Semantic Indexing (CL-LSI)

The model CL-LSI [217] is to construct a **multi-lingual semantic space** using **Latent Semantic Indexing (LSI)**. LSI is a technique that applies **sin-**

gular value decomposition (SVD) to a **term-document matrix** to reduce its dimensionality and capture the latent semantic relationships among terms and documents. SVD is a mathematical operation that decomposes a matrix into three matrices:

$$A = U\Sigma V^T \tag{2.21}$$

where A is the term-document matrix, U is the term-concept matrix, Σ is the diagonal matrix of singular values, and V is the document-concept matrix.

Specifically, it utilizes a technique known as Latent Semantic Indexing (LSI) to reduce the dimensionality of a comprehensive matrix of word-document frequencies through singular value decomposition (SVD). By doing so, the model captures latent semantic relationships between words and documents across languages. The extension of LSI, termed Cross-Language LSI (CL-LSI), allows the model to handle multilingual corpora effectively. This entails concatenating word-document matrices from different languages into a unified matrix, subsequently applying LSI to generate a multilingual semantic space. Similar methods or advancements of the present model are observed in other papers, such as [91], [92], and [176].

2.2.4.(iii) Information-Theoretic Framework for Cross-Lingual Language Model (InfoXLM)

This model was built using information-theory concept hence named as **InfoXLM**. This model [191] aligns with cross-lingual word embedding, aiming to establish a shared representation space for words across languages. This cross-lingual model is jointly trained by maximizing the lower bounds of three different objective functions for gaining mutual informa-

tion. These three are (i) Monolingual token-sequence mutual information ($\mathcal{L}_{\text{MMLM}}$), (ii) cross-lingual token-sequence mutual information ($\mathcal{L}_{\text{TLM}}$), and (iii) cross-lingual sequence-level mutual information ($\mathcal{L}_{\text{CL}}$). The model uses both **monolingual** and **parallel** corpora to jointly train tasks, in the context of Multilingual Masked Language Modeling (MMLM), the task revolves around predicting masked tokens within the same language context, aiming to maximize the mutual information between the masked tokens and their corresponding context. The equation of the model is as follows:

$$\mathcal{L} = \mathcal{L}_{\text{MMLM}} + \mathcal{L}_{\text{TLM}} + \mathcal{L}_{\text{CL}}, \tag{2.22}$$

where $\mathcal{L}_{\text{MMLM}}$, $\mathcal{L}_{\text{TLM}}$, and $\mathcal{L}_{\text{CL}}$ are the losses of the MMLM, TLM, and CL tasks, respectively. This work experimentally demonstrated significant performance improvements over prior methods [192]. Some other works [181, 176, 175], siliar to this, proposed different cross-lingual pre-training models by making few modifications to InfoXLM model. Some other works, with few modifications in terms of the pre-training tasks and model architecture, leverage large-scale monolingual and parallel corpora to learn generalized representations for multiple languages.

2.3 EMBEDDING MODELS FOR NON-CATEGORICAL DATA

Non-categorical data embedding models are techniques that aim to map non-categorical data into a continuous vector space, ensuring that data points with similar features are represented by similar vectors in the embedding space. This category has witnessed various research efforts aimed at developing different models to capture as much of the non-categorical data's semantic information as possible.

We classify the embedding models used for learning data representations into three types: (i) Probabilistic Embedding, which considers the uncertainty and variability of data, including Bayesian, variational, and generative models, (ii) Predictive Model, which accounts for the dependency and correlation of data, including neural network, attention, and transformer models, and (iii) Graph Embedding model, which encapsulates the structural and relational properties of data, including graph neural network, graph convolutional network, and graph attention network models.

2.3.1 Predictive Models

This models have emerged as a powerful tool for capturing semantic and syntactic relationships between words. These models, unlike count-based models, learn word representations by predicting words in a given context, often leveraging neural networks or other machine learning methods [97]. One of the most popular predictive models is Word2Vec, which uses either a continuous bag-of-words (CBOW) or a skip-gram architecture to learn word embeddings from large corpora [109]. FastText is an extension of Word2Vec that incorporates subword information into word embeddings [141]. This feature allows FastText to handle out-of-vocabulary words and morphologically rich languages better than Word2Vec. In the context of IR, this can be particularly useful for dealing with queries or documents that contain rare or unseen words [164]. Another notable model is GloVe (Global Vectors for Word Representation), a hybrid model that combines the advantages of count-based and predictive models [113]. GloVe learns word embeddings from global word co-occurrence statistics, using a weighted least-squares objective function. This approach allows GloVe to leverage both local context information and global statistical information, resulting

in robust and high-quality word embeddings.

2.3.1.(i) hierarchical log-bilinear model (HLBL)

The Hierarchical Log-BiLinear (HLBL) [216] model uses a binary tree of words to represent the vocabulary and to predict the next word based on its context using a log-bilinear model. The equation of the model is

$$P(w\_n = w|w\_1 : n-1) = \prod_i P(d\_i|q\_i, w\_1 : n-1), \tag{2.23}$$

where w_n is the next word, $w\_1 : n-1$ is the context, d_i is the ith digit in the binary code for word w, and q_i is the feature vector for the ith node in the path from the root to the word w. The probability of each decision is given by

$$P(d_i = 1|q\_i, w\_1 : n-1) = \sigma(\hat{r}^T q\_i + b\_i), \tag{2.24}$$

where $\sigma(x)$ is the logistic function, $\hat{r}$ is the predicted feature vector for the next word computed as a linear combination of the context word feature vectors, and b_i is the bias for the node q_i.

2.3.1.(ii) Sentiment-enriched word vector learning

A vector space model [103] that learns word representations that capture both semantic and sentiment information. The key theory behind the model is that words that occur together in documents have similar semantic meanings, and words that are predictive of sentiment labels have similar sentiment orientations. The model uses a mix of unsupervised and supervised techniques to learn word vectors from a probabilistic document model and a logistic regression classifier. The equation of the model is

$$\max_{R,b,\psi,b_c} \sum_{k=1}^{|D|} \lambda ||\hat{\theta}_k||_2^2 + \sum_{i=1}^{N_k} \log p(w_i|\hat{\theta}_k; R, b) + \sum_{k=1}^{|D|} \frac{1}{|S_k|} \sum_{i=1}^{N_k} \log p(s_k|w_i; R, \psi, b_c) - \nu ||R||_F^2 \tag{2.25}$$

where R is the word representation matrix, b is the word bias vector, ψ is the logistic regression weight vector, b_c is the logistic regression bias scalar, D is the set of documents, $\hat{\theta}_k$ is the MAP estimate of the mixture variable for document k, w_i is the i-th word in document k, s_k is the sentiment label of document k, S_k is the set of documents with the same rounded value of s_k, and ν and λ are regularization parameters.

2.3.1.(iii) Global context-aware neural language model

The model [104] designed to learn word representations capturing both local and global context. The model leverages two neural networks to compute scores for local and global context, emphasizing word order and syntactic information in short sequences and semantic/topical information in the entire document. This approach enables the discrimination of the correct next word from random alternatives. Additionally, the model facilitates the learning of multiple representations for homonymous and polysemous words through context clustering and re-labeling occurrences. The equations of the model are

For local context score, score$_l$:

$$a_1 = f(W_1[x_1; x_2; \ldots; x_m] + b_1) \tag{2.26}$$

$$\text{score}_l = W_2 a_1 + b_2 \tag{2.27}$$

where $[x_1; x_2; \ldots; x_m]$ is the concatenation of the word embeddings

of the word sequence, f is an activation function, a_1 is the hidden layer activation, W_1 and W_2 are the network weights, and b_1 and b_2 are the biases.

For global context score, score_g:

$$c = \frac{\sum_{i=1}^{k} w(t_i) d_i}{\sum_{i=1}^{k} w(t_i)} \tag{2.28}$$

$$a_1^{(g)} = f(W_1^{(g)}[c; x_m] + b_1^{(g)}) \tag{2.29}$$

$$\text{score}_g = W_2^{(g)} a_1^{(g)} + b_2^{(g)} \tag{2.30}$$

where c is the weighted average of the document word embeddings, $w(t_i)$ is the idf-weighting function, $a_1^{(g)}$ is the hidden layer activation, $W_1^{(g)}$ and $W_2^{(g)}$ are the network weights, and $b_1^{(g)}$ and $b_2^{(g)}$ are the biases.

For final score, score:

$$\text{score} = \text{score}_l + \text{score}_g \tag{2.31}$$

2.3.1.(iv) Skip-gram model

In the paper [109], Skip-gram model uses the word vectors which are trained to predict the surrounding words in a sentence or a document, using a softmax function, a hierarchical softmax, or a negative sampling technique. The equation of the model is

$$\max_{v, v'} \frac{1}{T} \sum_{t=1}^{T} \sum_{-c \le j \le c, j \neq 0} \log p(w_{t+j} | w_t) \tag{2.32}$$

where v and v' are the input and output vector representations of words, w_t is the t-th word in the training sequence, c is the size of the training context, and $p(w_{t+j}|w_t)$ is the probability of a word given another

word, computed by one of the methods mentioned above.

Decomposes the term-question-answer tensor X into a sum of F rank-one tensors using outer product notation.

2.3.1.(v) CNN-LSTM

A deep learning-based architecture model [186] is to combine TF-IDF weighted Glove word embedding with CNN-LSTM architecture. The model consists of five layers, Weighted embedding layer transforms the input text into a matrix of word vectors, where each word is represented by a fixed-length vector. The word vectors are weighted by TF-IDF measure, which reflects the importance of each word in the text and the corpus. Convolution layer applies a stack of convolution filters with different sizes (1-g, 2-g, and 3-g) to the word vector matrix, and produces a set of feature maps that capture local patterns and n-gram features in the text. Max-pooling layer reduces the dimensionality of the feature maps by taking the maximum value of each subregion, and preserves the most salient features for each filter. LSTM layer processes the pooled feature maps using a recurrent neural network with long short-term memory units, which can capture long-term dependencies and sequential information in the text. Dense layeris a fully connected neural network that takes the output of the LSTM layer and produces the final output of the model, which is the sentiment polarity of the input text.

2.3.1.(vi) Adagrad-optimized Skip Gram Negative Sampling (A-SGNS)

The model [197] utilizes Adagrad-optimized Skip Gram Negative Sampling (A-SGNS) to construct a vector space model (VSM) for word representation. A-SGNS, optimized by Adagrad, captures word context from extensive text corpora, addressing rare words. Cosine similarity is

then computed between a given topic and web page terms from the word embedding matrix, forming a feature vector input for the RNN. The RNN, adept at sequential data and long-term dependencies, utilizes RMSprop for training, adjusting learning rates and preventing gradient issues. The RNN classifies web pages as relevant or irrelevant based on the feature vector and output weight, with the latter serving as a linear function mapping the hidden state to a probability value (0 to 1) indicating web page relevance to the specified topic. The equations of the model is

$$w_{rec} = w_{rec} - \frac{\omega}{\sqrt{E[g^2] + \epsilon}} g \tag{2.33}$$

Here, w_{rec} represents the recurrent weight, ω is the learning rate, $E[g^2]$ is the mean square of the gradient, ϵ is a small constant, and g is the gradient of the cost function with respect to w_{rec}.

2.3.1.(vii) Semantic Information Extraction

The model [125] is a way to improve word embeddings by using information extraction techniques to discover semantic relationships between words from different sources, such as definitions, synonyms, and lists. The methodology is that some contexts exhibit more semantically meaningful information than others, and that these contexts can be used to adjust the word embeddings trained by the word2vec model. The equation of the model is:

$$\mathcal{L} = -l \cdot \log f - (1 - l) \cdot \log(1 - f) \tag{2.34}$$

where $f = \sigma(v_{w_t}^T \cdot v_{w_r})$ and l is the label (1 for positive samples and 0 for negative samples).

2.3.1.(viii) Distributional hypothesis

This model [201] which states that words appearing in similar context tend to have similar meaning. The paper uses two popular word embedding models CBOW model and skip-gram model, to implement the technique. The CBOW model predicts the current word based on its context, while the skip-gram model predicts the surrounding words given the current word.

$$p(w_j|w_i) = \frac{\exp(v_{w_j}^T v_{w_i})}{\sum_{w \in W} \exp(v_w^T v_{w_i})} \tag{2.35}$$

where v_w is the vector representation of the word w, and W is the vocabulary of all words.

2.3.1.(ix) Set-Based Word Vector Similarity

The model [137] is an approach to represent queries and documents as sets of embedded word vectors, and to compute the similarity between them using a mixture of Gaussians model. In this approach, words in documents or queries are encoded as real-valued vectors, capturing their semantic nuances through techniques like word2vec. To refine document representations, the set of word vectors is clustered into K topics using the K-means algorithm, each represented by a cluster centroid.

This not only reduces dimensionality and noise but also encapsulates the document's primary themes. The similarity between a query and a document is then gauged by the posterior query likelihood, linked to the average similarity between query word vectors and the document's cluster centroids. This comprehensive approach integrates word embeddings, clustering, and probabilistic modeling to enhance semantic understanding and theme extraction in document-query relationships. The query likelihood of

a document given a query is given by:

$$P(d|q) = \alpha P_{LM}(d|q) + (1-\alpha)P_{WVEC}(d|q) \tag{2.36}$$

where α is a parameter that controls the relative weight of the text-based query likelihood $P_{LM}(d|q)$ and the word vector-based query likelihood $P_{WVEC}(d|q)$.

2.3.1.(x) Relevance-Based Word Embedding

The model [151] is a technique for learning high-dimensional dense vector representations for words based on the notion of relevance, instead of term proximity. In this model, the words that are observed in the documents relevant to a particular information need are similar to each other. Two neural network-based models for learning relevance-based word embeddings. The objective is to maximize the likelihood of generating relevance model probabilities for the whole training set.

$$p_b(w|q;\theta_R) = \frac{\exp(\mathbf{w}^T\mathbf{q})}{\sum_{w' \in V}\exp(\mathbf{w'}^T\mathbf{q})} \tag{2.37}$$

where $\mathbf{w}$ and $\mathbf{q}$ are the word and query embedding vectors, respectively, V is the vocabulary set, and θ_R is the set of parameters to be learned.

Relevance posterior estimation model (RPE) casts the problem of estimating the relevance distribution as a binary classification task: Given a pair of word w and query q, does w come from the relevance distribution of q The model estimates the relevance posterior probability using a sigmoid function. The objective is to minimize the cross-entropy loss function using noise contrastive estimation.

$$p_b(R = 1|\mathbf{w}, \mathbf{q}; \theta_R) = \frac{1}{1 + e^{-\mathbf{w}^T\mathbf{q}}} \tag{2.38}$$

where R is a Boolean variable indicating whether the word-query pair comes from the relevance distribution or not.

2.3.1.(xi) Sentiment-Specific Word Embedding (SSWE)

The model [114] is a way of learning continuous representations for words and phrases that capture the sentiment information of texts. This model is to integrates the sentiment polarity of sentences into the loss functions of three neural networks, which are trained with massive distant-supervised tweets selected by positive and negative emoticons. The equations of the model SSWEr, which relaxes the hard constraints in SSWEh and uses a ranking objective function, the hinge loss of SSWEr is modeled as:

$$\text{loss}_r(t) = \max(0, 1 - \delta_s(t) f_r^0(t) + \delta_s(t) f_r^1(t)) \tag{2.39}$$

where f_r^0 is the predicted positive score, f_r^1 is the predicted negative score, $\delta_s(t)$ is an indicator function reflecting the sentiment polarity of a sentence.

2.3.2 Probabilistic Word Embedding

Probabilistic Word Embedding is a technique that aims to represent words as distributed vectors in a continuous vector space. The basic model introduced in this field is the Latent Semantic Analysis (LSA) [200]. This model uses a matrix factorization approach to capture the latent semantic relationships between words and represents them as vectors. However, the basic LSA model has some inherent challenges. One of the challenges is the reliance on co-occurrence statistics, which can be affected by data sparsity and noise. Another challenge is the lack of interpretability of the resulting embeddings.

Over the years, several papers have proposed innovative techniques to overcome the challenges of the basic LSA model and enhance probabilistic word embedding models. For example, the GloVe model [113] introduces a weighted least squares objective function that balances the capturing of global word co-occurrence statistics and local context windows. This approach addresses the issue of data sparsity and noise in co-occurrence statistics. Another approach is the Skip-gram Negative Sampling (SGNS) model [201], which aims to improve the efficiency and scalability of probabilistic word embeddings by training with negative samples. The SGNS model captures both syntactic and semantic relationships between words.

Probabilistic word embedding models have found applications in various natural language processing tasks. For instance, LSA has been utilized in information retrieval systems to enhance document ranking and retrieval precision [202]. GloVe embeddings have been used in sentiment analysis [203] and named entity recognition [204]. SGNS embeddings have shown promising results in machine translation [205] and question answering [206].

2.3.2.(i) Latent variable language model (LVLM)

In the paper [101], they described an approach that leverages unlabeled text to construct probabilistic models that capture the contextual information surrounding each word. The essence of the method lies in providing informative features to a supervised sequence labeler, thus enhancing its performance. The equation of the model is:

$$P(x|y) = \prod_i P(x_i|y_i)P(y_i|y_{i-1}) \tag{2.40}$$

where x is a word sequence, y is a latent state sequence, $P(x_i|y_i)$ is the emission probability, and $P(y_i|y_{i-1})$ is the transition probability. A recent

advancement of this model is observed in [157], which introduces a deep contextualized word representation that models both complex characteristics of word use and how these uses vary across linguistic contexts.

2.3.2.(ii) Neural Probabilistic Language Model

The model [97] uses a neural network to compute conditional probabilities for the next word based on previous ones, facilitating simultaneous learning of word feature vectors and network parameters. This approach marks a notable advancement in statistical language modeling by leveraging distributed representations for nuanced understanding of word relationships and context. The equation of this model is:

$$f(i, w_{t-1}, ..., w_{t-n+1}) = g(i, C(w_{t-1}), ..., C(w_{t-n+1})) \tag{2.41}$$

here, $f(i, w_{t-1}, \ldots, w_{t-n+1})$ signifies the likelihood of the subsequent word being i given the prior $n-1$ words $w_{t-1}, \ldots, w_{t-n+1}$. The function g is a neural network that transforms an input sequence of feature vectors into a probability distribution over words. $C(i)$ represents the feature vector corresponding to the word i. The parameter n denotes the model's order, reflecting the number of words considered in the contextual analysis.

2.3.2.(iii) Elliptical Probabilistic Embeddings (PWE)

The authors use Elliptical Probabilistic Embeddings (PWE) [215] to represent each word as a probability measure with a mean vector and a covariance matrix. The similarity between two words is computed using the Wasserstein-Bures pseudo-dot-product, which is an extension of the standard dot product to the space of elliptical measures. The authors train the PWE using the continuous bag of words (CBOW) paradigm, which

maximizes the similarity between the target word and the words in its context window, while minimizing it for words outside the context. The authors adapt the MatchPyramid model, a NIR architecture, to use PWE as the input word representations. They replace the cosine similarity with the Wasserstein-Bures normalized dot product to compute the matching matrix between query and document terms.

$$\frac{[\mu_{a,A} : \mu_{b,B}]}{\sqrt{2 + \mathrm{Tr}(A) + \mathrm{Tr}(B)}} = \frac{||a|| \cdot ||b||}{\sqrt{2 + \mathrm{Tr}(A) + \mathrm{Tr}(B)}} \tag{2.42}$$

They also propose an extension of the MatchPyramid model that combines FastText and WordNet embeddings to leverage both character n-grams and semantic relations information.

2.3.2.(iv) Gumbel-box process

The paper [188] is based on Represent words as n-dimensional axis-aligned hyperrectangles, or boxes, in a latent space. A box probability model, where the probability of a word is proportional to the volume of its box, and the joint probability of two words is proportional to the volume of their intersection. The expected volume of a box or an intersection is approximated by

$$\mathbb{E}[\max(Y - X, 0)] \approx \beta \log(1 + \exp(\frac{Y - X}{\beta} - 2\gamma)) \tag{2.43}$$

where X and Y are Gumbel random variables, β is the scale parameter, and γ is the Euler-Mascheroni constant.

2.3.2.(v) Auto-encoding variational Bayes (AEVB)

A stochastic variational inference and learning algorithm for directed probabilistic models with continuous latent variables [107] . In model, each word is generated from a latent variable z that follows a prior distribution $p(z)$. The conditional distribution of each word given z is $p(x|z;\theta)$, where x is the one-hot vector representation of the word and θ are the generative model parameters. A variational approximation to the intractable posterior distribution of z given x, which is $q(z|x;\phi)$, where ϕ are the variational parameters. The paper uses a multivariate Gaussian distribution with a diagonal covariance matrix for $q(z|x;\phi)$, and a neural network to compute the mean and variance of $q(z|x;\phi)$ from x. Derive a lower bound on the marginal likelihood of the text corpus, which is

$$L(\theta,\phi;X) = E_{q(z|x;\phi)}[logp(x|z;\theta) + logp(z) - logq(z|x;\phi)] \tag{2.44}$$

This lower bound can be estimated by sampling z from $q(z|x;\phi)$ and using a reparameterization trick to make it differentiable with respect to θ and ϕ. Optimize the lower bound with respect to θ and ϕ using stochastic gradient ascent, where the gradients are computed by backpropagation through the neural network encoder and decoder. The resulting word embeddings are the mean vectors of $q(z|x;\phi)$ for each word x in the vocabulary.

2.3.3 Graph Word Embedding

Graph Word Embedding is a technique that aims to capture the structural and relational properties of graphs, such as knowledge graphs, social networks, and citation networks, by representing nodes as vectors that depend on their neighbors and edges. In the provided source, the section on

Graph Word Embedding explores different models that have been developed in this field. One of the basic models introduced is the Graph2Vec [153], which uses matrix factorization techniques to learn graph vectors that represent the content and structure of graphs. Another model discussed in this section is the GraphSAGE (Graph Sample and Aggregated) [207], which leverages a neighborhood aggregation strategy to learn graph embeddings.

The basic models in graph word embedding face certain challenges. One challenge is handling the dynamic and evolving nature of language and information in graphs. To overcome these challenges, researchers have proposed innovative methodologies. For example, the GraphSAGE model incorporates an inductive learning framework, allowing it to generalize to unseen nodes and graphs during the embedding process [207]. The Graph2Vec model has been enhanced with techniques like node2vec, which employs random walks to capture both local and global structural information in graphs [208].

Graph word embedding models find applications in various domains. For instance, Graph2Vec has been used in network-based recommendation systems [153] and social network analysis [209]. GraphSAGE has been successfully applied in tasks such as link prediction [207] and node classification [210]. These models have demonstrated their effectiveness in capturing the underlying patterns and relationships present in graph data.

Recent advancements in graph word embedding focus on addressing specific challenges and further enhancing the performance of these models. For instance, the Graph Attention Network (GAT) [211] introduces a graph attention mechanism that allows for more fine-grained modeling of node dependencies. Another approach, the Graph convolutional network (GCN) [212], utilizes a simplified version of spectral graph convolutions to capture

graph structure information more efficiently.

In conclusion, graph word embedding techniques such as Graph2Vec, GraphSAGE, GAT, and GCN have significantly contributed to the field of graph representation learning. These models have been applied in various domains, showing promise in tasks such as recommendation systems, social network analysis, and link prediction. Ongoing research aims to further enhance the capabilities of graph word embedding models by addressing challenges related to dynamic and evolving graph data, and by incorporating more sophisticated attention mechanisms and convolutional operations.

2.3.3.(i) vec2graph

The methodology in the paper [170] is to visualize word embeddings as graphs, where words are nodes and edges are determined by cosine similarities between words. The paper also introduces a Python library called vec2graph, which can produce interactive graphs from any pre-trained word embedding model. The paper uses the following equation to convert cosine similarities x_{sim} into weights on graph edges:

$$\begin{aligned} y_{inv} &= 1 - x_{sim} \\ w_{magn} &= y_{inv} \times 100 \\ z_{dist} &= w_{magn} \times \log(w_{magn})) + 10 \end{aligned} \tag{2.45}$$

where y_{inv} is a cosine distance, the 10 constant is the radius of graphic circles representing nodes, and z_{dist} is the final distance value piped to the graph layout algorithm as a weight on the corresponding edge (the distance between the nodes).

2.3.3.(ii) metapath2vec

The model metapath2vec [149] is a framework for learning latent representations of nodes in heterogeneous networks, based on meta-path-guided random walks and heterogeneous skip-gram.

A meta-path scheme P that specifies the types of nodes and relations to be traversed by a random walker in the heterogeneous network G. Generate meta-path-based random walks from each node in G, following the transition probability defined by P. Use the heterogeneous skip-gram model to learn the node embeddings X that maximize the probability of predicting the heterogeneous context of each node, given its meta-path-based neighborhood. Use heterogeneous negative sampling to optimize the skip-gram model efficiently, by specifying one set of multinomial distributions for each type of nodes in the output layer. The equation for the heterogeneous skip-gram model is:

$$p(c_t|v;\theta) = \frac{e^{X_{c_t} \cdot X_v}}{\sum_{u_t \in V_t} e^{X_{u_t} \cdot X_v}} \tag{2.46}$$

where c_t is the context node of type t, v is the target node, X is the node embedding matrix, V_t is the set of nodes of type t, and θ is the model parameter.

2.3.3.(iii) GraphGlove

The paper [193] used learning graph word embeddings, where each word is a node in a weighted graph and the distance between words is the shortest path distance between the corresponding nodes. A task-specific loss function is used based on the GloVe objective, either using the graph distance or the dot product between distance vectors. Initialize the graph

with a subset of edges, using nearest neighbors and random words from the Euclidean GloVe embedding space. Learn the edge weights and probabilities by minimizing the loss function with L0 regularization and stochastic gradient descent. Use the learned graph to measure word similarity and analogy by computing the shortest path distance or the correlation of distance vectors.

2.3.3.(iv) GraphWave

The model GraphWave [162], which stands for *Graph Wavelets via Heat Diffusion* is a method that learns structural embeddings of nodes in graphs based on spectral graph wavelets and empirical characteristic functions. For each node in the graph, compute a spectral graph wavelet that captures the diffusion pattern of a heat kernel centered at that node. This wavelet is a function of the eigenvalues and eigenvectors of the graph Laplacian. Treat the wavelet coefficients as a probability distribution over the graph and embed them into a low-dimensional space using the empirical characteristic function, which is a complex-valued function that summarizes all the moments of the distribution. The equation that defines the spectral graph wavelet for node a is:

$$\Psi_a = U\text{diag}(g_s(\lambda_1), \ldots, g_s(\lambda_N))U^T\delta_a \tag{2.47}$$

where U is the matrix of eigenvectors of the graph Laplacian $L = D - A$, $\lambda_1 < \lambda_2 \leq \cdots \leq \lambda_N$ are the eigenvalues of L, g_s is the heat kernel filter with scaling parameter s, and δ_a is the one-hot vector for node a.

2.3.3.(v) Graph Attention Networks (GATs)

The paper [196] proposes a novel word embedding method based on graph attention networks (GATs). The node features of the word graph are initialized as one-hot vectors. The paper uses a four-layer GAT to encode the word graph, where the first three layers are attention layers and the last layer is a masked word node model. The paper uses the cross-entropy loss function to train the model, and evaluates the word embeddings on text classification tasks.

2.3.3.(vi) Word-Graph2vec

Word-Graph2vec [199], a Graph-based word embedding algorithm, which converts a large corpus into a word co-occurrence graph, then samples word sequences from this graph by random walk, and trains the word embedding on these sequences using the skip-gram model. The word co-occurrence graph is a weighted directed graph, where each node represents a unique word, each edge represents the co-occurrence of two words, and each edge weight represents the number of times the two words appear together in the same order in the corpus. The node weight represents the importance of the word in the corpus and is calculated by either term frequency (TF) or term frequency-inverse document frequency (TF-IDF).

2.4 LITERATURE REVIEW OF RERANKING

Our main goal is to enhance reranking techniques so that search outcomes are more relevant and accurate. Our primary objective is to enhance reranking methodologies to improve the accuracy and relevance of search results. To fulfill this objective, we conducted an extensive review of var-

ious ranking and reranking models. We started with the ranking model, i.e Sparse Bi-Encoder model [21] a first-stage ranker, which utilizes explicit sparsity regularization and a log-saturation effect on term weights. This model is improved in paper [24] which includes modifications to the pooling mechanism, which is the benchmarking of a model based solely on document expansion. Inspired by the sparsity regularization in Sparse Bi-Encoder, our model incorporates Dropout regularization to enhance its performance.

Another ranking model, Dense Bi-Encoder model [25] has demonstrated its effectiveness in capturing complex semantic relationships between texts as shown in figure 2.1. It utilizes a dual-encoder framework that indexes passages in a low-dimensional, continuous space for efficient retrieval. The paper [27] further enhances the DenseBiEncoder by incorporating unsupervised pre-training, which enables the model to leverage large corpora for improved performance. Additionally, [28] proposes a novel pre-training architecture architecture addresses the fragility to training data noise and the need for large batches. By considering this, we utilized large batches for tokenizing query-document pairs as input for the scoring mechanism in our model.

The paper [7] introduces T2Ranking, a comprehensive Chinese benchmark for passage ranking, designed to address the limitations of existing datasets which are predominantly English-based and lack fine-grained relevance annotations(figure 2.2). T2Ranking consists of over 300K queries and 2M unique passages, annotated with 4-level graded relevance scores by expert annotators. The study assesses the dataset using widely-used ranking models, highlighting the difficulty of T2Ranking and the room for advancement in passage ranking algorithms.

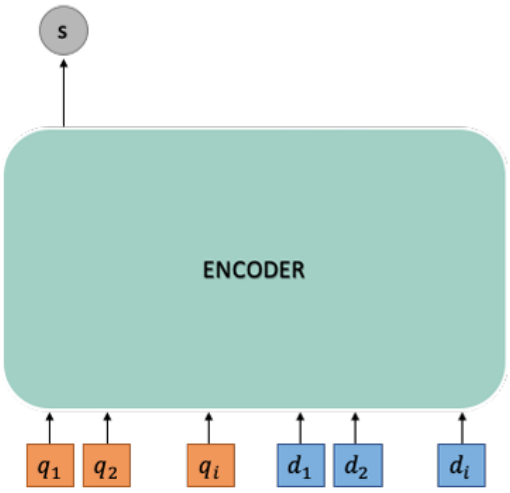

Figure 2.3: Direct deep LMs rerankers

The paper [22] presents TILDE (Term Independent Likelihood moDEl), a novel BERT-based model for passage re-ranking in information retrieval systems(figure 2.3). TILDE addresses the inefficiency of deep language models by pre-computing term likelihoods during indexing, eliminating the need for inference at query time. The paper also introduces a bi-directional training loss, BiQDL, which maximizes both query and document likelihoods simultaneously, enhancing the model's performance. This paper [23] presents a novel adaptation of a pretrained sequence-to-sequence model, specifically the T5 model, for the task of document ranking. Unlike traditional classification-based ranking approaches that utilize encoder-only transformer architectures like BERT, this technique creates relevance labels as "target words" using a sequence-to-sequence model. The logits of these target words are read as relevance probabilities for ranking, and the model is adjusted to generate the words "true" or "false" to signify document relevance.

The paper [29] introduces an attention mechanism into the traditional CNN model, enabling it to focus on more relevant parts of input sentences.

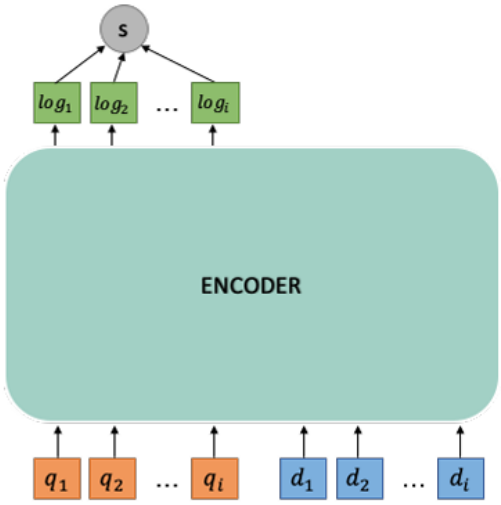

Figure 2.4: Deep query likelihood models

By giving distinct weights to the various sections of the sentence according to their importance to the categorization goal, this invention enhances performance. On the other hand, [30] demonstrates the effectiveness of CNNs when applied to pre-trained word vectors from word2vec, using a single layer of convolution and max-over-time pooling. The paper [32] delves into using RNNs highlighting the advantages of multi-task learning. This approach allows the model to learn and perform multiple tasks concurrently, thereby enhancing its ability to generalize and handle complex classification problems.

This paper [26] introduces a novel approach by leveraging large pre-trained generative models, such as GPT2 and BART, for document ranking tasks. The authors propose a method that fine-tunes a global language model on the task of query generation conditioned on document content. They also explore the use of unlikelihood losses to improve IR performance by penalizing negative examples. This paper [31] presents a novel framework known as Deep Contextualized Term Weighting (DeepCT). The DeepCT architecture creates context-aware term weights for passage retrieval by utilizing the

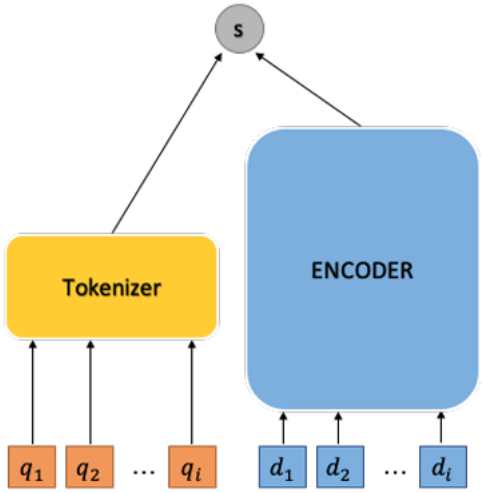

Figure 2.5: TILDE

transformer encoder of BERT to gather contextual aspects of words. Unlike traditional term frequency-based methods, DeepCT accounts for the semantic and syntactic roles of terms within their specific textual contexts. By utilizing this method, it is possible to create document-specific term weights that can be effectively retrieved using popular bag-of-words models such as BM25. These weights can then be stored using a regular inverted index.

The paper [33] first introduces the use of cross-encoders in zero-shot retrieval scenarios, showcasing their robustness and generalization capabilities. The CrossEncoder method in paper [34] is used to improve document ranking by incorporating multidimensional relevance statements into the cross-encoder re-ranking process. This approach enriches the representation of documents, leading to more accurate ranking results. Additionally, The paper [35] proposes an adaptive retrieval and indexing strategy for k-nearest neighbour (k-NN) search using cross-encoders. By refering all the models, we contributed to the modified cross encoder architecture in the above mentioned ways and additionally by calculating the cosine similarity score loss and sigmoid activation score loss separately, combining them to

reduce the final loss of the model during training.

2.5 LITERATURE REVIEW OF RERANKING USING QUERY EXPANSION

A variety of IR algorithms have been developed over the past decades, but some gaps remain in handling the variability of language and queries[47]. While baseline algorithms like tf-idf [48] remain widely used, they can struggle with language variability and lack native relevance ranking[49]. More recent neural [50] and non-parametric Bayesian algorithms [51] aim to address these issues through statistical language modeling [52] and manifold ranking [53]. Bayesian models offer several advantages over baseline models like tf-idf, including better statistical language modeling, uncertainty estimation [54], flexibility, adaptability, and the ability to perform manifold ranking. These advantages make Bayesian models important in improving relevance ranking and handling language variability in information retrieval tasks. However, comparative evaluation of these algorithms on large real-world datasets has been limited.

Query rewriting [55], [56], [57], [58] and query expansion algorithms have emerged as powerful tools in the field of IR to address the challenges of language variability and relevance ranking. Query rewriting algorithms, such as the one proposed by [59], focus on adapting the search query itself to bridge the gap between the input text and the needed knowledge in retrieval. They introduce a new framework, Rewrite-Retrieve-Read, for retrieval-augmented Large Language Models (LLMs), where an LLM is prompted to generate the query, which is then used to retrieve contexts. A trainable rewriter is used to cater to the black-box LLM reader.

On the other hand, query expansion techniques aim to improve re-

trieval performance by expanding the original query with additional relevant terms [60]. This process of reformulation helps to overcome the problem of synonymy and polysemy, thus increasing the chances of matching the user's query to the representations of relevant ideas in documents [61]. These techniques have found applications in various fields of IR, including personalized social document retrieval, question answering, cross-language IR, information filtering, and multimedia IR [61]. Despite their effectiveness, comparative evaluation of these algorithms on large real-world datasets remains a challenge.

Chapter 3

DATASETS

3.1 DATASET USED IN WORD EMBEDDINGS

Dataset	Resource
PTB dataset	https://catalog.ldc.upenn.edu/docs/LDC95T7/cl93.html
WT2 dataset	https://b2find.dkrz.de/dataset/2e904cd9-3780-5bdc-807d-617b18572106
W103 dataset	https://bonndata.uni-bonn.de/file.xhtml?persistentId=doi:10.60507/FK2/LFS5VY/WGQX5V&version=1.0
WordSim-353	https://gabrilovich.com/resources/data/wordsim353/wordsim353.html
SimLex-999	https://fh295.github.io/simlex.html
Stanford Rare Word Similarity Dataset	https://nlp.stanford.edu/~lmthang/morphoNLM/
XNLI	https://github.com/facebookresearch/XNLI
MLQA	https://github.com/facebookresearch/MLQA
Brown Corpus	https://varieng.helsinki.fi/CoRD/corpora/BROWN/
Flickr30K	https://huggingface.co/datasets/nlphuji/flickr30k
MNIST	https://www.tensorflow.org/datasets/catalog/mnist
Frey Face	https://rdrr.io/github/jlmelville/snedata/man/frey_faces.html
AMiner CS dataset	https://cn.aminer.org/data

Table 3.1: Datasets and Resources

The datasets used by the models described above are lsited in a table 3.1 with their available resources and description of the datasets are explained. The German GUR350 dataset contains 350 German word pairs, each annotated with a similarity score. It serves as a valuable resource for evaluating the performance of word embeddings and algorithms designed for measuring semantic similarity in the German language. The Penn Treebank dataset, or PTB dataset, consists of over 4.5 million annotated words from various sources such as Wall Street Journal articles. It is widely used for training and evaluating language models and parsers in English. The WikiText-2 (WT2) dataset is a collection of over 2 million words extracted from Wikipedia articles, providing a diverse and challenging dataset for language modelling tasks. The W103 dataset, a subset of WordNet, contains 103 semantic classes, making it suitable for tasks involving semantic similarity and relatedness. These datasets play a crucial role in advancing research

in natural language processing by providing standardized benchmarks and datasets for evaluation.

The WordSim-353 dataset comprises 353 word pairs, each annotated with a similarity score ranging from 0 to 10, commonly used for evaluating word embeddings' ability to capture semantic similarity. SimLex-999 consists of 999 word pairs with similarity scores also ranging from 0 to 10, specifically designed to assess lexical semantics capture. The Stanford Rare Word Similarity Dataset focuses on rare words, providing a specialized evaluation set with similarity scores ranging from 0 to 10 for such vocabulary. The Part-of-speech tagging dataset contains annotated texts where each word is labeled with its part of speech, serving as a benchmark for evaluating part-of-speech tagging models' accuracy in assigning the correct part-of-speech tag to each word in a sentence or text.

XNLI, or Cross-lingual Natural Language Inference, consists of 5,000 sentence pairs in 15 languages, resulting in 75,000 annotated pairs. Each pair is labeled with one of three textual entailment relations: entailment, contradiction, or neutral. MLQA, or Multilingual Question Answering, comprises over 5,000 question-answer pairs in eight languages, totaling 7,793 context-question-answer triples. Tatoeba is a vast dataset containing over 8 million sentences in multiple languages, offering translations for many sentences across different languages. The Brown Corpus includes text from 500 sources, categorized into 15 genres, with a total of one million words. It serves as a fundamental resource for linguistic research and is commonly used for training and evaluating natural language processing models.

The AP News Corpus contains over 1.9 million news articles from the Associated Press, spanning a wide range of topics and dates, making

it a valuable resource for natural language processing tasks such as text classification and information retrieval. Flickr30K consists of 30,000 images, each paired with five English captions, totaling 150,000 captions, making it suitable for tasks like image captioning and multimodal learning. MNIST comprises 60,000 training images and 10,000 test images of handwritten digits, each image being a 28x28 pixel grayscale image, widely used as a benchmark for training and testing machine learning models for image recognition. Frey Face contains a set of 1,400 face images, each 20x28 pixels, commonly utilized for tasks like facial recognition and image processing.

The Word Analogy Test Set is a dataset consisting of word analogy questions, typically in the form of "A is to B as C is to D," used to evaluate word embeddings' ability to capture semantic relationships. The AMiner CS dataset contains metadata from over 3.8 million computer science research papers, including author information and citation data, often utilized for author identification and citation analysis tasks. The SCWS dataset comprises 2,003 word pairs with human-annotated similarity scores, focusing on contextual word similarity to evaluate word embeddings. The BATS dataset consists of 10,000 analogy questions divided into four types: inflectional morphology, derivational morphology, lexicographic semantics, and encyclopedic semantics, providing a more comprehensive evaluation of word embeddings compared to the Word Analogy Test Set.

3.2 DATASET USED IN RERANKING

Based on document collections, queries, and their pertinent and irrelevant documents, the dataset is split into three distinct datasets: the Document_Collections Dataset, the Train_Queries Dataset, and the Train_Triplets Dataset. Details about those datasets are provided in the paragraphs that

follow.

Document_Collections Dataset

This dataset consists of documents which cover a range of topics and are structured textual information. The `doc_id` serves as a unique identifier for each document, while the `doc_text` column contains the actual text content of the document. Many tasks related to natural language processing, including text summarization, document categorization, and information retrieval, can be performed using this dataset.

Train_triplets Dataset

This dataset as shown in figure 4.1, consists of 96,923 query-document pairs. Each row corresponds to a single query (`query_id`) and two documents: a positive document (`pos_doc`) and a negative document (`neg_doc`). While the negative document is deemed irrelevant, the positive document is thought to be pertinent to the question. The arrangement of the dataset makes it easier to assess retrieval algorithms according to how well they can prioritize relevant items over irrelevant ones for a particular query.

Train_queries Dataset

This dataset as shown in the figure 4.2 contains a total of 966 rows, each representing a unique query (`query_id`) and its corresponding text (`query`). These queries are likely sourced from various information retrieval scenarios, such as search engine queries, question-answering systems, or other text-based information retrieval tasks.

1	query_id	pos_doc	neg_doc
2	2150	3606712	2136007
3	2150	3606712	6175864
4	2150	3606712	1955393
5	2150	3606712	5337331
6	2150	3606712	161660
7	2150	3606712	4252873
8	2150	3606712	7965427
9	2150	3606712	957797
10	2150	3606712	4550464
11	2150	3606712	8540520
12	2150	3606712	1102965
13	2150	3606712	6510254
14	2150	3606712	3255548
15	2150	3606712	5163446
16	2150	3606712	6873952
17	2150	3606712	4350241
18	2150	3606712	6541657
19	2150	3606712	1134437
20	2150	3606712	6143876
21	2150	3606712	6470724
22	2150	3606712	4123484
23	2150	3606712	901889
24	2150	3606712	3920923
25	2150	3606712	3135986

Figure 3.1: Train Triplets Dataset Sample

1	query_id	query
2	2150	Avery name meaning
3	6494	Transverse is a term that refers to
4	6915	What are the symptoms of Lyme disease
5	6925	What band names were the Beatles known as
6	8074	Where does flint come from
7	9005	_____ can be used by cells to store energy, form biological membranes, and serve as chemical messengers.
8	9538	a.o.smith stock price
9	9766	abbreviation sis definition in odp
10	10818	active exhalation respiratory definition
11	11490	address for davidson fine arts school in augusta ga
12	11722	admission definition
13	12255	aerugo definition
14	13395	age most children take begin walking
15	15581	all about iq
16	15744	allergy symptoms in children
17	17126	amount of caffeine in death wish coffee
18	17754	analog definition electronics
19	19998	apparent temperature means a combination of
20	22723	are granulocytes neutrophils
21	23563	are masks required when caring for droplet precaution patients
22	24336	are platelets pieces of cells
23	25207	are survivor benefits taxable income
24	26145	are water meters read remotely
25	26463	argentina's flag meaning
26	26561	arizona diamondbacks food

Figure 3.2: Train Queries Dataset Sample

3.3 DATASET USED IN RERANKING USING QUERY EXPANSION

In this experimentation, we utilized an extensive Microsoft Academic Knowledge Graph [66], which encompasses metadata of more than 240 million scholarly articles spanning various academic disciplines. From this comprehensive resource, we USED two datasets encompassing elements such as article titles, summaries, citations, key terms, associated academic fields, dates of publication, event locations, as well as details on authors and their institutional connections. These two datasets were named Computer Science data and Physics data.

3.4 DATASET USED IN TEXT CLASSIFICATION

The AG's news topic classification dataset is a collection of more than 1 million news articles gathered from over 2000 news sources by the academic news search engine ComeToMyHead, which has been active since July 2004. This dataset is made available to the academic community for research purposes in various fields, including data mining, information retrieval, and data compression. The dataset used for text classification benchmarking contains 4 largest classes selected from the original corpus, with each class consisting of 30,000 training samples and 1,900 testing samples. In total, there are 120,000 training samples and 7,600 testing samples. The file classes.txt contains a list of classes corresponding to each label. This dataset is utilized as a benchmark in the paper "Character-level Convolutional Networks for Text Classification" by Xiang Zhang, Junbo Zhao, and Yann LeCun, presented at the Advances in Neural Information Processing Systems 28 (NIPS 2015) conference.

Chapter 4

TWO STAGE PASSAGE RANKING

This paper introduces a novel CrossEncoder architecture as in 4.1, which helps in the field of information retrieval by proficiently encoding pairs of query and document text . Unlike traditional models, it employs a pre-trained transformer model for sequence classification, designed to discern between relevant and irrelevant document pairs. The scoring mechanism implemented in the cross-encoder, not only extracts logits corresponding to the positive class but furthermore calculates the cosine similarity among the query and document's [CLS] token embeddings. This dual-scoring measure, unique to our model, enhances its capacity to order documents according to how relevant they are to a certain inquiry. The forward method, computes the loss during training, using a combined loss function that captures the divergence between the predicted scores/similarities and the true labels. This method, coupled with dropout regularization, ensures the model's robustness against overfitting. By minimizing this combined loss, the CrossEncoder model learns to effectively discriminate between relevant and irrelevant documents. This novel approach sets our model apart from base models, marking a significant leap forward in the quest for improved search result quality and user satisfaction. Our major contributions in this research work are as follows:

- It proposes a modified Cross Encoder for a two-stage passage ranking in IR

- It proposes a new loss function by combining cosine similarity and sigmoid function.

- It presents the experimental analysis on three different datasets to evaluate the performance of proposed model.

4.1 MODEL ARCHITECTURE

The CrossEncoder architecture as shown in the figure 3.1, is designed to jointly encode pairs of query and document text using a pre-trained transformer model. Learning a scoring formula which can distinguish between relevant and irrelevant document pairs for any given query is its main goal. The architecture comprises three main components: the pre-trained transformer model, the scoring mechanism, and the loss function. The CrossEncoder architecture is presented in Algorithm 1, and its functionality is illustrated in the Figure 4.1. Our contributions to the CrossEncoder code include adding a sigmoid activation function to the logits in the scoring method, implementing cosine similarity calculation between the [CLS] token representations of the query and document in the scoring method, and adding dropout regularization to the model. The architecture is explained in the following paragraphs.

The first step in the CrossEncoder architecture is the initialization phase, where a pre-trained transformer model for sequence classification is loaded. This model is set up for binary classification, with two labels representing positive (relevant) and negative (irrelevant) pairs. Additionally, a cross-entropy loss function and a dropout layer are initialized for training and regularization purposes, respectively.

The scoring mechanism takes a batch of tokenized query-document

pairs as input. The pairs are passed through the pre-trained transformer model, and the logits (raw scores) corresponding to the positive class are extracted. The sigmoid activation function is then used to translate these logits into normalized scores ranging from zero to one, which indicates the likelihood that a pair is positive (relevant). Additionally, the method computes the cosine similarity between the [CLS] token embeddings of the query and document, providing an alternative scoring measure based on the similarity of their representations in the transformer's embedding space.

The forward method takes two inputs, 'pos_pairs' and 'neg_pairs', which are batches of pairs of sequences. The method first calls the 'score_pairs' method to compute the scores and cosine similarity for both positive and negative pairs. The scores are obtained by passing the input pairs through the pre-trained model and applying the sigmoid activation function to the logits. The cosine similarity is calculated by comparing the [CLS] token embeddings from the last hidden state of the model. The method then concatenates the scores and cosine similarity scores along the batch dimension, effectively combining the two types of scores. Next, it applies dropout regularization to the concatenated scores and cosine similarity scores to prevent overfitting. Subsequently, the technique generates labels for both positive and negative pairs, designating positive pairs as One and negative pairs as zero. The labels are concatenated along a new dimension to match the shape of the scores and cosine similarity scores. Finally, the method calculates the total loss as the sum of the CrossEntropyLoss applied to the scores and cosine similarity scores with the corresponding labels, effectively optimizing the model to predict the correct labels for the input pairs.

The training process is used to optimize the CrossEncoder model to assign higher scores to positive (relevant) query-document pairs and lower

scores to negative (irrelevant) pairs. The model can be used in retrieval or ranking tasks where the most relevant documents need to be surfaced for a user's query by learning to successfully discriminate between relevant and irrelevant documents for a given query by minimizing the combined loss.

Algorithm 1 CrossEncoder

```
model ← AutoModelForSequenceClassification.from_pretrained(model_name, num_labels=2)
loss ← CrossEntropyLoss()
dropout ← Dropout(dropout_rate)
for each pair in pairs do
    outputs ← model(**pair, return_dict=True)
    logits, cls_embeddings ← outputs.logits, outputs.hidden_states[-1][:, 0, :]
    scores ← sigmoid(logits[:, 1])
    query_embeddings ← cls_embeddings[::2]
    doc_embeddings ← cls_embeddings[1::2]
    cosine_scores ← cosine_similarity(query_embeddings, doc_embeddings)
    scores ← sigmoid(logits[:, 1])
end for
for each pos_pair in pos_pairs and neg_pair in neg_pairs do
    pos_scores, pos_cosine_scores ← score_pairs(pos_pair)
    neg_scores, neg_cosine_scores ← score_pairs(neg_pair)
    scores ← concatenate(pos_scores, neg_scores, dim=0)
    cosine_scores ← concatenate(pos_cosine_scores, neg_cosine_scores, dim=0)
    scores ← dropout(scores)
    cosine_scores ← dropout(cosine_scores)
    labels ← concatenate(ones(pos_scores.size()), zeros(neg_scores.size()), dim=1)
    loss ← loss(scores, labels) + loss(cosine_scores, labels)
end for
```

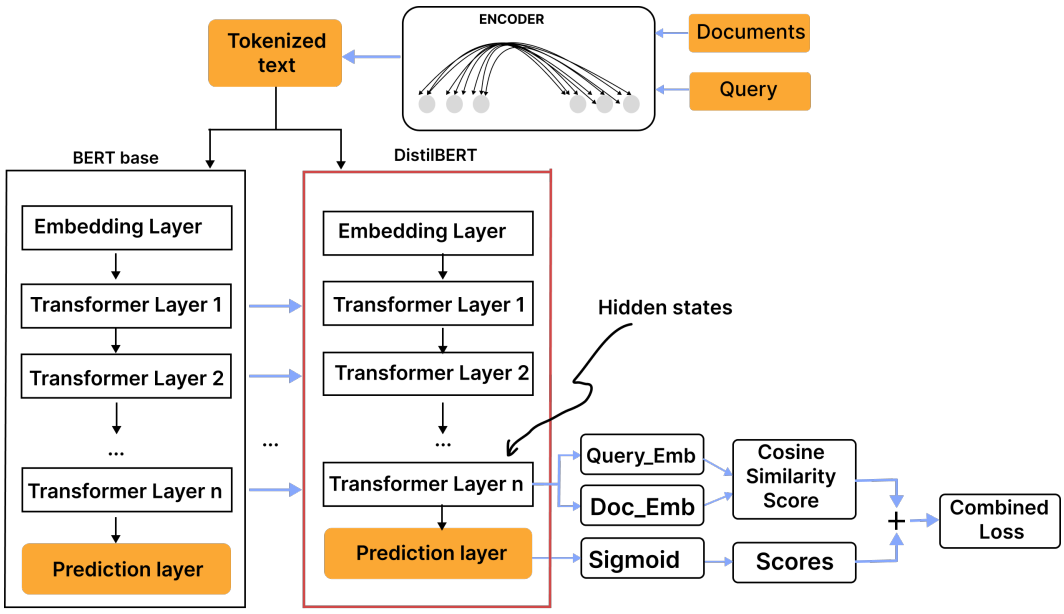

Figure 4.1: Modified CrossEncoder Architecture

4.2 EXPERIMENTATION

4.3 PRETRAINED LANGUAGE MODEL

The encoder architecture incorporates a pre-trained language model as the initial processing layer. Specifically, we employ DistilBERT [36], a distilled version of the BERT model trained on large text corpora. Using a distilled model(figure 4.3) reduces computational requirements while retaining substantial language knowledge. DistilBERT transforms input text to contextually encoded token embeddings via multiple self-attention layers, capturing key linguistic features and interactions vital for natural language understanding. The pretrained weights provide a strong initialization for the model to build upon rather than learning from scratch. Additionally, the language model embeddings are fine-tuned during the triplet training process, adapting the general-domain representations towards the specifics

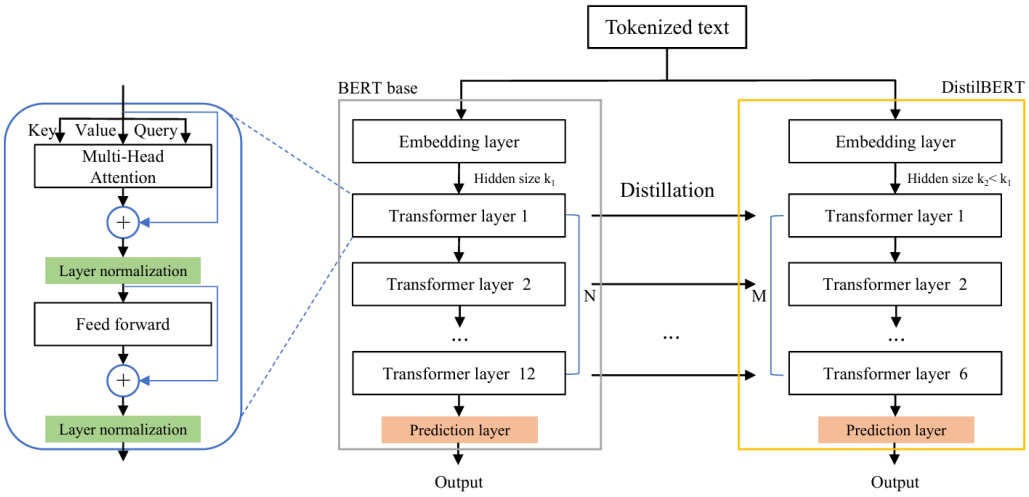

Figure 4.2: DistilBERT Model Architecture

of information retrieval for further improvements. The pretraining and fine-tuning integrate external knowledge while maintaining flexibility to specialize in semantic matching. Overall, the pre-trained DistilBERT encoder provides an essential first stage of the model to leverage extensive context and language understanding, offering a solid foundation for subsequent convolutional and recurrent network layers tailored to relevance matching.

4.3.1 Tokenization

We utilize a RoBERTa as shown in the figure 4.4, pre-trained model from the Hugging Face's transformers library for tokenization purpose. The pre-trained model serves a crucial role in encoding both the queries and documents into dense vector representations. This encoding process involves feeding the text of the queries and documents into the model, which subsequently outputs a high-dimensional vector for each piece of text. These vectors encapsulate the semantics of the text, making them highly valuable for downstream tasks such as document ranking.

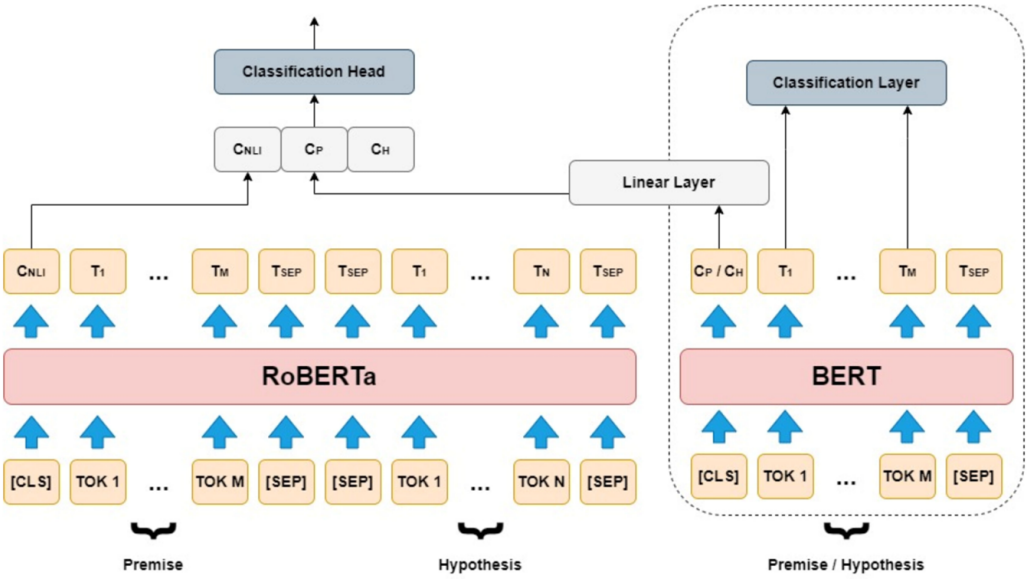

Figure 4.3: RoBERTa Model Architecture

The encoding method accepts a batch of (query, document) pairs, tokenizes them using the Hugging Face's tokenizer, and then feeds them into the pre-trained model. The model then outputs a vector for each pair. The second element of the logits is utilized as the score for the pair. This score is a representation of the model's estimate of the relevance of the document to the query.

4.3.2 First-Stage Ranker

In the first stage of our project, we employed a ranker to generate an initial ranking of documents for each query. This ranker is based on traditional retrieval methods or modern approaches like BM25, Dense BiEncoders or Sparse BiEncoders. The purpose of this first-stage was to narrow down the potentially large document collection to a more manageable subset of relevant documents. This stage ranking process served as a critical foundation for the subsequent Cross Encoder reranking stage, setting the stage for more refined and accurate ranking of the documents. It's a testa-

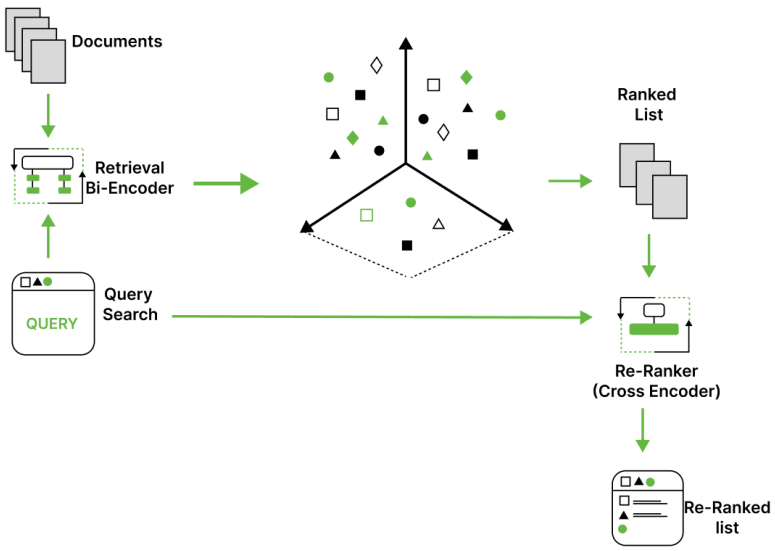

Figure 4.4: Two Stage Ranking Architecture

ment to the power of combining traditional and modern ranking methods in a two-stage ranking framework.

4.3.3 Two-Stage Ranker

In the second stage of our project, we implemented a reranker shown in the figure 4.5, to refine the initial ranking of documents generated by the first-stage ranker. This reranker utilizes advanced neural network models, such as Cross Encoders, to reevaluate the relevance of documents based on more complex features and interactions. By considering a wider range of factors, including semantic relationships and contextual information, the reranker aims to provide a more accurate ranking of documents for each query. This second-stage reranking process builds upon the initial ranking from the first stage, enhancing the overall effectiveness of our two-stage ranking framework. Furthermore, our ranking architecture, depicted in Figure 5.1, illustrates the flow of the reranking process within our framework.

4.3.4 Hyperparameters and Training

The Cross Encoder model in our neural IR system is trained using backpropagation and gradient-based optimization with the Adam update rule. The model leverages pre-trained language model embeddings, which are fine-tuned during training to enhance performance. A uniform distribution in the interval [-0.1, 0.1] is randomly sampled to initialize the remaining model parameters.

The hyperparameters that yield the best performance on the validation set are selected for the final evaluation. In the absence of a validation set, we resort to 10-fold cross-validation. The final hyperparameters are as follows: The embedding size from the pre-trained language model is 768. The hidden layer size for the Cross Encoder is also 768, matching the embedding size. The initial learning rate is set to 5e-5, following a linear decay schedule. L2 weight decay regularization of 0.01 is applied to all parameters, excluding biases and layer normals.

Using a contrastive loss function on triplets of the query, positive document, and negative document, the model is trained to convergence. The trained Cross Encoder model can then be utilized to encode query and document texts into vectors for semantic search and ranking.

4.3.5 Training Data

We generate training triplets by sampling queries from training topics along with their judged relevant and non-relevant documents.Documents that are pertinent are considered positives, whereas those that are not are considered negatives. Instead of mining random negatives, we use the initial retrieval scores from unsupervised approaches such as BM25 to mine hard negatives, ensuring the model learns from demanding instances. The

triplets are batched and fed to our model during training.

4.3.6 Training Loss

The model is trained using triplet loss on query, positive document, negative document triplets. Given an anchor query q, a positive document d^+ that is relevant to the query, and a negative document d^- that is irrelevant, the loss function is defined as:

$$L = \max(0, \sigma - S(q, d^+) + S(q, d^-)) \tag{4.1}$$

where σ is a margin hyperparameter and $S(q, d)$ is the similarity score among the query q and document d embeddings generated by our model. Minimizing the loss has the effect of pulling the positive document embedding closer to the query in the semantic space while pushing the negative embedding further away. The margin hyperparameter σ controls how far negatives should be separated from positives. Larger margins result in better separation but can lead to slower convergence. Optimizing across many query-document training triplets enables learning text representations tailored for semantic search.

The model parameters are updated using Adam optimizer to minimize the total triplet loss over the batch. We use standard practices like learning rate warm-up and decay to ensure stable optimization. Overfitting is avoided by early halting based on retrieval evaluation indicators on the validation set. The model that performs the best during validation is saved.

4.4 EXPERIMENTAL ENVIRONMENT

We utilized the powerful NVIDIA V100 GPU, equipped with 640 Tensor Cores, providing 130 teraFLOPS (TFLOPS) of deep learning performance. This GPU enabled us to achieve a remarkable 12X Tensor FLOPS for deep learning training and 6X Tensor FLOPS for deep learning inference. To leverage this hardware, we conducted our experiments in Google Colab, a cloud-based environment seamlessly integrated with the V100.

4.5 RESULTS

The ModifiedCrossEncoder significantly outperformed other models across all metrics which can be seen in Table 5.1. Compared to DenseBiEncoder, it increased P@10 by 0.02, nDCG@10 by 0.18, AP@100 by 0.18, and RR@100 by 0.18. Gains over SparseBiEncoder were modest, around 0.01-0.03 increase. However, versus the standard CrossEncoder, ModifiedCrossEncoder showed substantial boosts - 0.03 higher P@10, 0.26 higher nDCG@10, 0.27 higher AP@100, and 0.26 higher RR@100.

Model	**P@10**	**nDCG@10**	**AP@100**	**RR@100**
BM25	0.06	0.63	0.54	0.59
DenseBiEncoder	0.07	0.59	0.56	0.57
SparseBiEncoder	0.08	0.74	0.71	0.72
CrossEncoder	0.06	0.51	0.47	0.49
ModifiedCrossEncoder	0.09	0.77	0.74	0.75

Table 4.1: Performance of different models

In contrast, the ModifiedCrossEncoder model exhibits resilience in correctly ranking documents for various queries, as it achieves a balanced performance across all metrics. These findings demonstrate the promise of transformer-based designs such as the ModifiedCrossEncoder for improving the accuracy of document retrieval tasks in the field of information retrieval.

This model's robust performance highlights its potential usefulness in real-world applications.

4.6 DISCUSSION

This two-stage ranking framework combines traditional and modern ranking methods effectively and efficiently. The first stage uses a BiEncoder (BM25, Dense, or Sparse) to generate an initial ranking of documents for each query, reducing the collection to a relevant subset. The second stage employs a CrossEncoder to re-rank documents by jointly encoding query document pairs into a transformer's [CLS] vector, then calculating relevance scores through a linear layer. The proposed Modified CrossEncoder model demonstrated superior performance compared to the DenseBiEncoder, SparseBiEncoder, and CrossEncoder models.When comparing Modified CrossEncoder to the other four encoders, notable trends emerge across various evaluation metrics. Starting with Precision at 10 (P@10), Modified CrossEncoder demonstrates superiority over BM25, DenseBiEncoder, and CrossEncoder by 0.03, 0.02, and 0.03, respectively.

However, it slightly trails SparseBiEncoder by 0.01, indicating competitive but not yet optimal precision within the top 10 search results. In normalized Discounted Cumulative Gain at 10 (nDCG@10), Modified CrossEncoder show cases significant improvement over BM25, DenseBiEncoder, and CrossEncoder, surpassing them by 0.18, 0.18, and 0.26, respectively. Nevertheless, it falls behind SparseBiEncoder by 0.03, suggesting room for optimization to match the comprehensiveness of top-ranked results. Moving to Average Precision at 100 (AP@100), Modified CrossEncoder outperforms BM25, DenseBiEncoder, and CrossEncoder by 0.20, 0.18, and 0.27, respectively. Despite this, it lags slightly behind SparseBiEncoder by 0.03,

indicating the need for enhancements to achieve comparable comprehensiveness in captureing relevant documents. Considering Reciprocal Rank at 100 (RR@100), Modified CrossEncoder demonstrates significant improvements over BM25, DenseBiEncoder, and CrossEncoder by 0.16, 0.18, and 0.26, respectively. However, it still trails SparseBiEncoder by 0.03, suggesting further refinement opportunities in prioritizing highly relevant documents. The findings of this research highlight the promise of transformer based designs for improving the accuracy of document retrieval tasks. The robust performance of the Modified CrossEncoder model underscores its potential usefulness in real-world applications.

Chapter 5

RANDOMNESS IN QUERY EXPANSION

In the realm of Information Retrieval (IR), the quest for more informative and relevant search results drives the exploration of novel techniques. One such approach, rooted in information theory, is the method proposed by Gianni Amati (GA), which emphasizes terms specific to feedback documents to enhance the retrieval process. This method provides a robust mathematical framework for measuring term informativeness, crucial for improving the accuracy of search results. Implementing this approach using PyTerrier, we applied a class that implements the KL query expansion model from the Divergence from Randomness Framework. Our implementation, accompanied by detailed documentation and associated run scripts available on GitHub, ensures experimental results that are both repeatable and replicable. This approach not only enhances the effectiveness of information retrieval systems but also contributes to the broader goal of advancing IR methodologies

5.1 APPROACH

This section explains how Kullback-Leibler (KL) divergence in the context of information retrieval, which is a measure of how one probability distribution diverges from a second expected probability distribution is applied to enhance the precision of information retrieval. We discuss the implementation of KL divergence in query expansion and its role in addressing

the vocabulary mismatch problem. The section further explores the nuances of asymmetric KL divergence. By the end of this section, readers will gain a comprehensive understanding of the application and significance of KL divergence in improving information retrieval systems.

5.1.1 Binomial Distribution Approximation

The Binomial Distribution (BD) [65] is a type of discrete probability distribution in statistics. It describes the number of successes in a certain number of independent Bernoulli trials, each with the same probability of success. The probability and error of approximation of the binomial distribution are represented as p and ϕ respectively. This binomial distribution is approximated using an information theoretic divergence, denoted as d, from ϕ. When applying the binomial distribution to both term-weighting and query expansion, the definition of p and ϕ will satisfy the relation $p < \phi$ without loss of generality. The approximation of the binomial distribution using the asymmetric Kullback-Leibler divergence is particularly useful when the probabilities of term occurrence are very small. This is a common scenario in large document collections. This approach allows for more accurate and efficient processing of information retrieval tasks. Under the assumption of $p < \phi$, the contribution in the divergence,

$$D(\phi, p) = (1-p)\log_2(1-\frac{p}{\phi}) \tag{5.1}$$

where $D(\phi, p)$ is negative and also, both p and ϕ are very small and thus $\log_2(1-\frac{p}{\phi})$, which can be easily shown to be approximately $p-\phi$, is also close to 0. Therefore, it is straightforward to derive a further approximation of the

Bernoulli process by means of the asymmetric Kullback-Leibler divergence,

$$KL(\phi, p) = p \log_2 \frac{p}{\Phi} \tag{5.2}$$

The approximation of Bernoulli's process via the divergence function Formula can be rewritten as:

$$D(p_{\text{Eq}}, p_D) = p_{\text{Eq}} \cdot \log_2 \frac{p_{\text{Eq}}}{p_D} + (1 - f) \cdot \log_2 (1 - p_{\text{Eq}}) \frac{p_{\text{Eq}}}{(1 - p_D)} \tag{5.3}$$

where p_{Eq}, and p_D represent the frequencies of the term in the subset E_q and in the collection D respectively, where E_q is the elite set of the query and f is the frequency.

5.1.2 Query Expansion using Kullback-Leibler Divergence

The query expansion model can be primarily characterized by the use of the hypergeometric distribution. This distribution is the result of a sampling process where words are drawn from an urn and not replaced. Consider a population D consisting of TotFrD tokens, with F tokens representing the same word t. In the context of query expansion, E represents the highest-ranked documents in an initial document ranking. In the sample E, we observe FE tokens of the word t. The hypergeometric distribution calculates the probability $P(F_E|D, E)$ of observing exactly F_E tokens in the sample, assuming the sample was randomly selected. As E comprises the top-ranked documents, the hypergeometric distribution offers a measure of how much the sample deviates from randomness for a specific word.

$$\mathrm{Inf}_{\mathrm{Eq}}(t) = \mathrm{TotFr}_{E_q} D(p_{\mathrm{Eq}}, p_D) + \frac{1}{2}\log_2(2\pi \mathrm{TotFr}_{E_q}(1-p_{\mathrm{Eq}}))[Bi] \tag{5.4}$$

The above expression of the information constant does not depend on the term but on the size of the collection and the sample, the term-weights within the query are those from the binomial model up to a constant.

let us assume $c = \frac{1}{2}\log_2 \pi$, then:

$$\mathrm{Inf}_{\mathrm{Eq}}(t) = \mathrm{TotFr}_{E_q} D(p_{\mathrm{Eq}}, p_D) + \frac{\log_2 \mathrm{TotFr}_{E_q} + c}{2} + O\left(\frac{1}{\mathrm{TotFr}_{E_q}}\right) \tag{5.5}$$

Since $\frac{\log_2 \mathrm{TotFr}_{E_q}}{2}$ is independent of the term t, then its contribution in the sum is a constant and also $O\left(\frac{1}{\mathrm{TotFr}_{E_q}}\right)$ is approximation error constant. Therefore, the information content can be supposed to be proportional to:

$$\mathrm{Inf}_{\mathrm{Eq}}(t) \sim D(p_{E_q}, p_D) \tag{5.6}$$

Moreover, the contribution $\frac{(1-p_{E_q})\log_2(1-p_{E_q})}{p_D}$ in $D(p_{E_q}, p_D)$ is very small and negative, because we may in general assume that $p_{E_q} > p_D$. Thus, we derive a further approximation of the Bernoulli process:

$$\mathrm{Inf}(t|E_q) \sim p_{E_q} \cdot \log_2 \frac{p_{E_q}}{p_D}[KL] \tag{5.7}$$

which is the asymmetric Kullback-Leibler divergence.

The Kullback-Leibler divergence is a important of modern retrieval models, enabling them to discern the relevance of documents based on the divergence from randomness in term distributions.

5.2 EXPERIMENTAL SETUP

This section experimentally presents how the inclusion of the KL-divergence concept in query expansion can improve the IR model performance. For this, we implemented various base IR models with and without KL divergence and then evaluated them over two different IR datasets. The sequence of steps in our experiments, such as dataset preparation, preprocessing, indexing, query expansion, and re-ranking are presented in the following subsections.

5.2.1 Computational Infrastructure and Environment

In our research, we utilized the powerful NVIDIA V100 GPU, equipped with 640 Tensor Cores, providing an impressive 130 teraFLOPS (TFLOPS) of deep learning performance. This GPU enabled us to achieve a remarkable 12X Tensor FLOPS for deep learning training and 6X Tensor FLOPS for deep learning inference. To leverage this hardware, we conducted our experiments in Google Colab, a cloud-based environment seamlessly integrated with the V100.

5.2.2 Preprocessing

To obtain a high-quality and clean document collection, we parsed the RDF content and then selected unique English papers based on the language attribute. Later in this preprocessing pipeline, it standardized the text data by removing extra white spaces, punctuation, dots in acronyms, and hyperlinks. Later it replaces the symbols with their corresponding terms, for example "&" symbol is replaced with "and". To focus on informative words, later it removes the stop-words and then applies the stemming operation in which it replaces all words with their root words [67, 68]. These

Table 5.1: Corpus Statistics

Statistic	Physics Collection	CS Collection
Documents	49,26,753	4,809,674
Unique Terms	30,06,661	2,235,016
Postings	26,80,80,089	254,719,768
Tokens	41,27,89,547	415,150,364
Avg. Tokens per Doc	83.79	86.39

operations in the preprocessing pipeline increase the quality of the dataset and performance of our IR model.

5.2.3 Indexing

After preprocessing the collection of documents in two groups, Physics and Computer Science, we build an indexer. The main aim of this indexing is to retrieve the statistics of the pre-processed text corpus efficiently and then retrieve the relevant documents using these statistics in the IR model. We achieve this by iterating over each document in the collection, extracting the document's ID and text, and then indexing this information using library methods in the PyTerrier framework. This resulting index is stored and used later to efficiently retrieve the corpus statistics. The statistics derived from the indexers of the Physics and Computer Science corpus are given in second and third columns of Table 5.1. The Table 5.1 demonstrates the statistics like total number of documents, total number of unique terms, total number of postings, total number of tokens and average number of tokens per document in each group of collection.

5.2.4 Baseline Models Implementation

In this section, we present the baselines we evaluated. In Table 5.2, the performance of these models is presented. For instance, the TF_IDF model has a P@10 of 0.1664, R@10 of 0.1070, nDCG@10 of 0.1909, and RR@100 of

Table 5.2: Performance of Baseline Models

Model	Performance of Models					
	Computer Science			Physics		
	P@10	R@10	nDCG@10	P@10	R@10	nDCG@10
TF_IDF	16.64	10.70	19.09	21.64	10.82	23.98
BM25	16.56	10.67	18.98	22.16	11.09	24.87
DPH	15.84	10.12	18.10	20.62	10.35	23.42
DFRee	15.60	10.01	17.87	20.21	10.14	22.81
InL2	15.59	9.90	17.77	19.84	10.01	22.40
LGD	15.55	9.99	17.75	19.72	9.91	22.23
BB2	15.43	9.88	17.62	19.67	9.93	22.18
DLH	15.20	9.74	17.33	19.85	10.03	22.40
DFIC	15.13	9.68	17.36	20.09	10.09	22.80
PI2	14.75	9.42	16.80	18.92	9.54	21.42

0.4417. The TF model exhibits the lowest performance across all metrics due to its simplistic nature, which relies solely on term frequency without considering document length or inverse document frequency, while the model with the highest performance varies depending on the metric.

5.2.5 Query ReWriting

To enhance the performance of these baseline models, a query rewriting technique known as Kullback–Leibler (KL) Divergence is applied. KL Divergence is a measure of how one probability distribution diverges from a second, expected probability distribution. Table 5.3 presents the performance of the models after applying KL Divergence. The application of KL Divergence generally results in an increase in the performance metrics for each model. For example, the TF_IDF model now has a P@10 of 0.1751, R@10 of 0.1127, nDCG@10 of 0.1973, and RR@100 of 0.4366. This demonstrates the effectiveness of KL Divergence in enhancing the performance of baseline models in information retrieval tasks. The architecture of the model is shown in Fig. 5.1 We also tried another query rewriting technique called

Table 5.3: Performance of Baseline Models with KL Divergence

Model	Performance of Models with KL Divergence					
	Computer Science			Physics		
	P@10	R@10	nDCG@10	P@10	R@10	nDCG@10
TF_IDF	17.51	11.27	19.73	22.10	11.73	24.70
BM25	17.41	11.21	19.61	23.57	11.73	25.40
DFRee	16.83	10.76	18.93	21.56	10.82	24.05
LGD	16.73	10.76	18.93	21.12	10.63	23.48
DPH	16.58	10.57	18.61	21.76	10.92	24.27
InL2	16.38	10.49	18.40	20.79	10.46	23.14
DLH	16.04	10.29	17.97	20.84	10.48	23.21
DFIC	16.03	10.24	18.03	21.20	10.64	23.60
BB2	15.84	10.13	17.88	20.20	10.15	22.50
PI2	15.01	9.57	16.85	19.54	9.82	21.83

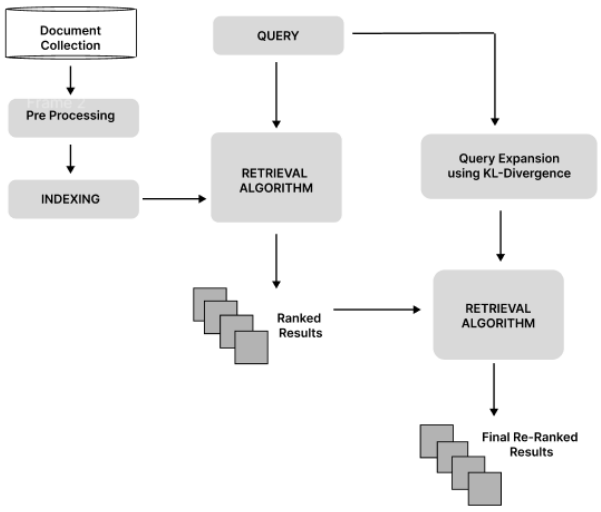

Figure 5.1: KL-Reranking

Table 5.4: Performance of Baseline Models with KNRM reranker

Name	P@10	R@10	nDCG@10	RR@100
TF_IDF	0.0319	0.0192	0.0317	0.1096
BM25	0.0327	0.0194	0.0313	0.1070
DFree	0.0251	0.0161	0.0239	0.1012
LGD	0.0211	0.0140	0.0221	0.0947
DPH	0.0195	0.134	0.0206	0.0929

tokenization. The Query Tokeniser approach doesn't change much. The performance of base models on the Query Tokeniser approach is less than the KL Divergence approach.

5.2.6 Neural Implementation

For the Neural Approach, we tried different methodologies. The first reranker used was KNRM: we used it with a Bert vocabulary. TABLE 5.4 shows the obtained results, compared to the base models, whose performances were not improved. The lack of improvement in performance for the base models when combined with KNRM may be attributed to the limited capacity of KNRM to effectively capture the complex relationships and semantics present in the documents and queries. For this task, we considered only the models TF_IDf, BM25, Dfree, Lgd, and Dph. The remaining models are showing lower results constantly.

The second attempt was to use a Vanilla-Bert reranker, but also in this case, the results in TABLE 5.5 were not as good as the base models.

5.3 RESULTS AND DISCUSSION

In conclusion, our experiments were conducted on a large-scale multi-domain benchmark dataset for Personalized Search and explored various

Table 5.5: Performance of Baseline Models with Vanilla-BERT reranker

Name	P@10	R@10	nDCG@10	RR@100
TF_IDF	0.0319	0.0192	0.0317	0.1096
BM25	0.0322	0.0194	0.0313	0.1070
DFree	0.0294	0.0175	0.0239	0.0981
LGD	0.0240	0.0139	0.0218	0.0969
DPH	0.0199	0.135	0.0196	0.0940

methodologies to enhance retrieval performance in the defined information retrieval tasks. Initially, we evaluated a set of baseline models (Table 5.2), where TF_IDF emerged as the top performer for Precision at 10 (P@10) with a score of 0.1664. However, its performance across other metrics displayed inconsistency. Subsequently, we introduced Kullback-Leibler (KL) Divergence for query rewriting, detailed in Table 5.3. Compared to the baseline models (Table 5.2), KL Divergence notably improved performance across all models and evaluation metrics. For instance, TF_IDF with KL Divergence achieved a P@10 of 0.1751, surpassing the baseline TF_IDF's P@10 of 0.1664. This underscores the effectiveness of KL Divergence in enhancing retrieval accuracy.

Our investigation also explored neural approaches, employing KNRM and Vanilla-BERT rerankers (Tables 5.4 and 5.5). However, these approaches failed to outperform either the baseline models or the KL Divergence approach. For instance, the KNRM reranker for TF_IDF resulted in a P@10 of 0.0319, significantly lower than the TF_IDF with KL Divergence (P@10 of 0.1751). This substantial difference highlights the superior performance of KL Divergence in this context, likely due to its ability to capture semantic relationships and improve query-document alignment.

Chapter 6

TEXT CLASSIFICATION

Text classification, a fundamental task in information retrieval, has undergone a paradigm shift with the advent of pre-trained models. Unlike traditional models that start from scratch, pre-trained models like BERT (Bidirectional Encoder Representations from Transformers) leverage vast corpora to enhance generalization and accuracy. BERT, a standout example in this domain, employs bidirectional context understanding and serves as the foundation for various architectural advancements. For instance, BERT CNN integrates Convolutional Neural Networks (CNNs) for localized pattern recognition, while BERT DPCNN utilizes Deep Pyramid CNNs (DPCNNs) for improved feature extraction. Additionally, BERT RNN incorporates Recurrent Neural Networks (RNNs) for capturing sequential dependencies. The Transformer architecture, introduced alongside BERT, has gained popularity for its ability to model long-range dependencies through self-attention mechanisms. These pre-trained models have significantly elevated performance in tasks such as sentiment analysis, topic categorization, and spam detection, ushering in a new era in information retrieval.

6.1 METHODOLODY

The proposed model is a sophisticated deep learning architecture that leverages the power of transformer models, convolutional neural networks (CNNs), recurrent neural networks (RNNs), and long short-term memory

(LSTM) networks. The model begins by inputting 'input ids' and 'attention mask' into a pretrained transformer model, which predicts masked tokens in the input sequence. The output from the transformer is then processed through a series of operations including softmax application, quantization, and embedding to yield dense vector representations.

These embeddings are further processed through multiple convolutional layers for feature extraction. Each convolutional layer is equipped with distinct filter sizes and applies a convolution operation and ReLU activation to the embeddings. The outputs of these layers are then max-pooled and concatenated to form the final feature representation.

This representation is subsequently fed into both an RNN and an LSTM layer for further processing. The final encoded output is formed by concatenating the last outputs of the RNN and LSTM layers. This resulting tensor represents the encoded sequences and can be further utilized for subsequent processing, such as feeding into a linear layer to obtain final predictions.

This model is designed to handle complex tasks and extract meaningful features from the input data. It combines the strengths of various neural network architectures to deliver robust and accurate predictions. For a detailed understanding of the model and its operations, please refer to the mathematical equations provided above. They offer a comprehensive view of the model's inner workings and the transformations applied to the data at each stage. This detailed view can help in understanding the model's functionality and its potential applications.

The process begins by inputting 'input_ids' and 'attention_mask' into a transformer model. This transformer is a pretrained masked language model, predicting masked tokens in the input sequence. The resulting

output from the transformer is represented as a tensor $L \in \mathbb{R}^{n \times v}$, where n is the sequence length and v is the vocabulary size.

Softmax and Quantization

The logits L are modified by zeroing out values corresponding to padded tokens (where the attention mask is False). A softmax function is then applied to these logits, converting them into probabilities. These probabilities are subsequently quantized into discrete values by multiplying with the vocabulary size and rounding off. The quantized probabilities are finally cast to the long datatype.

$$p = \text{softmax}(L') \quad \text{where} \quad p \in \mathbb{R}^{n \times v} \tag{6.1}$$

$$q = \text{round}(p \cdot (v - 1)) \quad \text{where} \quad q \in \mathbb{Z}^{n \times v} \tag{6.2}$$

$$q_{\text{long}} = \text{long}(q) \quad \text{where} \quad q_{\text{long}} \in \mathbb{Z}^{n \times v} \tag{6.3}$$

In summary, the logits L are first masked for padded tokens, followed by the application of a softmax function to obtain probabilities p. These probabilities are then quantized into integers q by multiplication with the vocabulary size and rounding off. Finally, q is converted to a long integer type q_{long}.

Quantized Probability Embeddings

The quantized probabilities are processed through an embedding layer to yield dense vector representations. These embeddings undergo reshaping and are then subjected to a Rectified Linear Unit (ReLU) activa-

tion function.

Quantized probability embeddings are obtained by the matrix multiplication of the long integer-type quantized probabilities q_{long} with the embedding weight matrix W. Let $W \in \mathbb{R}^{v \times d}$ be the embedding weight matrix, where d is the embedding dimension. The resulting tensor X is given by:

$$X = W \cdot q_{\text{long}} \tag{6.4}$$

Here, X belongs to $\mathbb{R}^{n \times d}$.

The embeddings X are reshaped by adding a singleton dimension. The tensor X' is obtained by:

$$X' = X.\text{unsqueeze}(1) \tag{6.5}$$

Now, X' belongs to $\mathbb{R}^{n \times 1 \times d}$.

The reshaped embeddings X' undergo the Rectified Linear Unit (ReLU) activation function to produce the output embeddings Y:

$$Y = \text{ReLU}(X') \tag{6.6}$$

Here, $Y = \max(0, X')$ and Y belongs to $\mathbb{R}^{n \times 1 \times d}$.

So, the quantized probabilities q_{long} are passed through an embedding layer with weights W to obtain embedded vectors X. These are reshaped into X' by adding a singleton dimension, and finally, a ReLU activation is applied to get the output embeddings Y.

Convolutional Layers

The embedded characters Y undergo convolutional feature extraction through multiple convolutional layers, each equipped with distinct filter sizes $f_1, f_2, \ldots, f_m$. For each filter i, the convolution operation and ReLU activation are applied to Y:

$$H_i = \text{ReLU}(\text{Conv}(Y; W_{f_i})) \tag{6.7}$$

Here, W_{f_i} denotes the weight matrix for filter size f_i, resulting in $H_i \in \mathbb{R}^{n \times 1 \times d}$. Subsequently, max pooling is performed on each H_i with a width of $(n - f_i + 1)$:

$$M_i = \text{MaxPool}(H_i) \tag{6.8}$$

The width of the max pooling operation is determined by $(n - f_i + 1)$, yielding $M_i \in \mathbb{R}^{(n-f_i+1) \times 1 \times d}$. The outputs of the max-pooled tensors $M_1, M_2, \ldots, M_m$ are concatenated along the first dimension to create Z:

$$Z = \text{Concat}(M_1, M_2, \ldots, M_m) \tag{6.9}$$

Here, $Z \in \mathbb{R}^{\sum(n-f_i+1) \times 1 \times d}$.

Finally, Z is reshaped to obtain O as the final feature representation:

$$O = \text{Reshape}(Z) \tag{6.10}$$

This results in $O \in \mathbb{R}^{-1 \times n \times md}$, providing the ultimate set of features for subsequent processing.

RNN and LSTM Layers

The output from the Convolutional Neural Network (CNN), denoted as O, is further processed through both a Recurrent Neural Network (RNN) and a Long Short-Term Memory (LSTM) layer. The final encoded output is formed by concatenating the last outputs of the RNN and LSTM layers.

$$\text{rnn\_in} = O^T \in \mathbb{R}^{n \times (-1 \times md)} \tag{6.11}$$

$$h_0 = 0 \in \mathbb{R}^{1 \times n \times p} \tag{6.12}$$

$$\text{rnn\_out}, = \text{RNN}(\text{rnn\_in}, h_0) \tag{6.13}$$

$$h_0 = 0 \in \mathbb{R}^{1 \times n \times q}, \quad c_0 = 0 \in \mathbb{R}^{1 \times n \times q} \tag{6.14}$$

$$\text{lstm\_out}, (\_) = \text{LSTM}(\text{rnn\_in}, (h_0, c_0)) \tag{6.15}$$

$$\text{encoded} = \text{Concat}(\text{rnn\_out}, \text{lstm\_out}) \tag{6.16}$$

where O is derived from the preceding section, p denotes the number of hidden units, $\text{rnn\_out} \in \mathbb{R}^{n \times p}$, q represents the number of LSTM hidden units, $\text{lstm\_out} \in \mathbb{R}^{n \times q}$, $\text{encoded} \in \mathbb{R}^{n \times (p+q)}$.

Thus, the transposed CNN output O serves as the input to both the RNN and LSTM layers, and the concatenated outputs form the final encoded representation. The resulting tensor represents the encoded sequences and can be further utilized for subsequent processing, such as feeding into a

linear layer to obtain final predictions.

6.2 EXPERIMENTATION RESULTS

Model	Chinese	English
Transformer	89.91%	83.69%
TextRCLSFM	90.54%	79.13%
TextRNN_att	90.90%	79.91%
Fastext_DPCNN	91.04%	76.46%
TextRNN	91.12%	77.64%
BERT_MR	91.12%	88.47%
TextCNN	91.22%	64.03%
BERT	91.22%	89.64%
DPCNN	91.25%	76.46%
TextRCNN	91.54%	78.59%
TextCRNN_Att	91.73%	80.46%
FastText	92.23%	73.29%
robert_DPCNN	94.44%	88.47%
bert_DPRCNN	**94.54**%	**91.89**%

Table 6.1: Model Performance

The Table 6.1 presents the performance of various models on Chinese and English datasets. The performance is measured in terms of accuracy, with higher percentages indicating better performance.

The Transformer model achieved an accuracy of 89.91% on the Chinese dataset and 83.69% on the English dataset. The TextRCLSFM model performed slightly better on the Chinese dataset with an accuracy of 90.54%, but its performance on the English dataset was lower at 79.13%. The TextRNN_att model showed a similar trend, with an accuracy of 90.90% on the Chinese dataset and 79.91% on the English dataset. The Fastext_DPCNN model achieved an accuracy of 91.04% on the Chinese dataset, but its performance dropped to 76.46% on the English dataset.

The TextRNN and BERT_MR models both achieved an accuracy of

91.12% on the Chinese dataset. However, the BERT_MR model outperformed the TextRNN model on the English dataset with an accuracy of 88.47%, compared to 77.64%. The TextCNN and BERT models both achieved an accuracy of 91.22% on the Chinese dataset. However, the BERT model significantly outperformed the TextCNN model on the English dataset, achieving an accuracy of 89.64% compared to 64.03%.

The DPCNN model achieved an accuracy of 91.25% on the Chinese dataset and 76.46% on the English dataset. The TextRCNN model performed slightly better on both datasets, with accuracies of 91.54% and 78.59% on the Chinese and English datasets, respectively. The TextCRNN_Att model achieved an accuracy of 91.73% on the Chinese dataset and 80.46% on the English dataset. The FastText model performed slightly better on the Chinese dataset with an accuracy of 92.23%, but its performance on the English dataset was lower at 73.29%. The robert_DPCNN model achieved an impressive accuracy of 94.44% on the Chinese dataset and 88.47% on the English dataset. The bert_DPRCNN model outperformed all other models, achieving the highest accuracies of 94.54% on the Chinese dataset and 91.89% on the English dataset.

Chapter 7

CONCLUSION

Our investigation into document retrieval techniques has provided valuable insights, particularly regarding the effectiveness of KL Divergence, which consistently outperformed baseline models and neural rerankers across various metrics. This highlights its potential for significantly enhancing retrieval performance, with future avenues for optimization and strategic combination with other techniques showing promise for even greater improvements. In evaluating different models, we found that the bert_DPRCNN model stood out as the top performer on both Chinese and English datasets, showcasing its versatility and robustness. However, the varying performance of models underscores the importance of selecting models based on task-specific requirements. For instance, simpler models like TextRNN or FastText may be more suitable when computational resources are limited, while more complex models like bert_DPRCNN offer unparalleled accuracy.Additionally, our study introduced a novel approach through a modified cross-encoder model specifically designed for document retrieval tasks. This model, which integrates traditional ranking methods like BM25 with modern transformer-based architectures, demonstrated significant improvements over baseline models across various evaluation metrics. While the sparse bi-encoder occasionally maintained a slight edge, the competitive performance of our modified cross-encoder highlights the promising potential of transformer-based models in enhancing retrieval accuracy. In conclusion, our research underscores the importance

of judiciously combining traditional and modern techniques to enhance information retrieval capabilities. By leveraging the power of KL Divergence for efficient retrieval, selecting models like bert_DPRCNN for optimal accuracy, and innovating with approaches like the modified cross-encoder, our findings pave the way for more effective and tailored document retrieval systems.

REFERENCES

[1] R. Nogueira and K. Cho, "Passage re-ranking with bert," arXiv preprint arXiv:1901.04085, (2019).

[2] E. Sheetrit, A. Shtok, and O. Kurland, "A passage-based approach to learning to rank documents," Information Retrieval Journal, vol. 23, pp. 159–186, 2020.

[3] V. Karpukhin, B. O guz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih, "Dense passage retrieval for open-domain question answering," arXiv preprint arXiv:2004.04906, (2020).

[4] D. L. Lee, H. Chuang, and K. Seamons, "Document ranking and the vector-space model," IEEE software, vol. 14, no. 2, pp. 67–75, (1997).

[5] R. Nogueira, Z. Jiang, and J. Lin, "Document ranking with a pretrained sequence-to-sequence model," arXiv preprint arXiv:2003.06713, (2020).

[6] G. Salton, J. Allan, and C. Buckley, "Approaches to passage retrieval in full text information systems," in Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, pp. 49–58, (1993).

[7] Xie, Xiaohui, et al. "T2ranking: A large-scale chinese benchmark for passage ranking." Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2023.

[8] M. Kobayashi and K. Takeda, "Information retrieval on the web," ACM computing surveys (CSUR), vol. 32, no. 2, pp. 144–173, (2000).

[9] O. Kolomiyets and M.-F. Moens, "A survey on question answering tech- nology from an information retrieval perspective," Information Sciences, vol. 181, no. 24, pp. 5412–5434, (2011).

[10] J. Goldstein, V. O. Mittal, J. G. Carbonell, and M. Kantrowitz, "Multi-document summarization by sentence extraction," in NAACL-ANLP 2000 workshop: automatic summarization, (2000).

[11] G. Salton, E. A. Fox, and H. Wu, "Extended boolean information retrieval," Communications of the ACM, vol. 26, no. 11, pp. 1022–1036, (1983).

[12] G. Salton, A. Wong, and C.-S. Yang, "A vector space model for automatic indexing," Communications of the ACM, vol. 18, no. 11, pp. 613–620, (1975).

[13] J. S. Whissell and C. L. Clarke, "Improving document clustering using okapi bm25 feature weighting," Information retrieval, vol. 14, pp. 466–487, (2011).

[14] M. A. Soliman and I. F. Ilyas, "Ranking with uncertain scores," in 2009 IEEE 25th international conference on data engineering. IEEE, pp. 317–328, (2009).

[15] X. Xie, Q. Dong, B. Wang, F. Lv, T. Yao, W. Gan, Z. Wu, X. Li, H. Li, Y. Liu et al., "T2ranking: A large-scale chinese benchmark for passage ranking," in Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, pp. 2681–2690, (2023).

[16] Y. Luan, J. Eisenstein, K. Toutanova, and M. Collins, "Sparse, dense, and attentional representations for text retrieval," Transactions of the Association for Computational Linguistics, vol. 9, pp. 329–345, (2021).

[17] R. Ren, S. Lv, Y. Qu, J. Liu, W. X. Zhao, Q. She, H. Wu, H. Wang, and J.-R. Wen, "Pair: Leveraging passage-centric similarity relation for improving dense passage retrieval," arXiv preprint arXiv:2108.06027, (2021).

[18] E. Choi, S. Lee, M. Choi, H. Ko, Y.-I. Song, and J. Lee, "Spade: Improving sparse representations using a dual document encoder for first-stage retrieval," in Proceedings of the 31st ACM International Conference on Information & Knowledge Management, pp. 272– 282, (2022).

[19] L. Wang, N. Yang, X. Huang, B. Jiao, L. Yang, D. Jiang, R. Majumder, and F. Wei, "Simlm: Pre-training with representation bottleneck for dense passage retrieval," arXiv preprint arXiv:2207.02578, (2022).

[20] R. Ren, Y. Qu, J. Liu, W. X. Zhao, Q. She, H. Wu, H. Wang, and J.-R. Wen, "Rocketqav2: A joint training method for dense passage retrieval and passage re-ranking," arXiv preprint arXiv:2110.07367, (2021).

[21] T. Formal, B. Piwowarski, and S. Clinchant, "SPLADE: sparse lexical and expansion model for first stage rank- ing," CoRR, vol. abs/2107.05720, (2021).

[22] Zhuang, Shengyao, and Guido Zuccon. "TILDE: Term independent likelihood moDEl for passage re-ranking." Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. 2021.

[23] Nogueira, Rodrigo, Zhiying Jiang, and Jimmy Lin. "Document ranking with a pretrained sequence-to-sequence model." arXiv preprint arXiv:2003.06713 (2020).

[24] T. Formal, C. Lassance, B. Piwowarski, and S. Clinchant, "Towards effective and efficient sparse neural information retrieval," ACM Trans. Inf. Syst., https://doi.org/10.1145/3634912, (2023).

[25] V. Karpukhin, B. Oguz, S. Min, L. Wu, S. Edunov, D. Chen, and W. Yih, "Dense passage retrieval for open-domain question answering," CoRR, vol. abs/2004.04906, https://arxiv.org/abs/2004.04906, (2020).

[26] Santos, Cicero Nogueira dos, et al. "Beyond [CLS] through ranking by generation." arXiv preprint arXiv:2010.03073 (2020).

[27] L. Gao and J. Callan, "Unsupervised corpus aware language model pre-training for dense passage retrieval," CoRR, vol. abs/2108.05540, https://arxiv.org/abs/2108.05540, (2021).

[28] J. C. Luyu Gao, "Is your language model ready for dense representation fine-tuning?" CoRR, vol. abs/2104.08253, 2021. [Online]. Available: https://arxiv.org/abs/2104.08253

[29] I. Alshubaily, "Textcnn with attention for text classifica- tion," CoRR, vol. abs/2108.01921, https://arxiv.org/abs/2108.01921, (2021).

[30] Y. Kim, "Convolutional neural networks for sentence classi- fication," CoRR, vol. abs/1408.5882, http://arxiv.org/abs/1408.5882, (2014).

[31] Dai, Zhuyun, and Jamie Callan. "Context-aware term weighting for first stage passage retrieval." Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval. 2020.

[32] P. Liu, X. Qiu, and X. Huang, "Recurrent neural network for text classification with multi-task learning," CoRR, vol. abs/1605.05101, http://arxiv.org/abs/1605.05101, (2016).

[33] G. Rosa, L. Bonifacio, V. Jeronymo, H. Abonizio, M. Fadaee, R. Lotufo, and R. Nogueira, "In defense of cross-encoders for zero-shot retrieval," (2022).

[34] R. Upadhyay, A. Askari, G. Pasi, and M. Viviani, "Enhancing documents with multidimensional relevance statements in cross-encoder re- ranking," (2023).

[35] N. Yadav, N. Monath, M. Zaheer, R. Fergus, and A. McCallum, "Adaptive retrieval and scalable indexing for k- NN search with cross-encoders," in The Twelfth International Conference on Learning Representations, https://openreview.net/forum?id=1CPta0bfN2, (2024).

[36] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, "Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter," arXiv preprint arXiv:1910.01108, (2019).

[37] Z. Dou, R. Song, and J.-R. Wen A large-scale evaluation and analysis of personalized search strategies pp. 581–590, Proceedings of the 16th international conference on World Wide Web, (2007).

[38] X. Shen, B. Tan, and C. Zhai Implicit user modeling for personalized search Proceedings of the 14th ACM international conference on Information and knowledge management, 2005, pp. 824–831.

[39] M. Speretta and S. Gauch Personalized search based on user search histories The 2005 IEEE/WIC/ACM International Conference on Web Intelligence (WI'05). IEEE, 2005, pp. 622–628.

[40] D. Carmel, N. Zwerdling, I. Guy, S. Ofek-Koifman, N. Har'El, I. Ronen, E. Uziel, S. Yogev, and S. Chernov Personalized social search based on the user's social network Proceedings of the 18th ACM conference on Information and knowledge management, 2009, pp. 1227–1236.

[41] P. Ferragina and A. Gulli A personalized search engine based on web-snippet hierarchical clustering Special interest tracks and posters of the 14th international conference on World Wide Web, 2005, pp. 801–810.

[42] Z. Fu, K. Ren, J. Shu, X. Sun, and F. Huang Enabling personalized search over encrypted outsourced data with efficiency improvement IEEE transactions on parallel and distributed systems, vol. 27, no. 9, pp. 2546–2559, 2015.

[43] J. Teevan, S. T. Dumais, and E. Horvitz Personalizing search via automated analysis of interests and activities Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, 2005, pp. 449–456.

[44] C. Carpineto and G. Romano A survey of automatic query expansion in information retrieval Acm Computing Surveys (CSUR), vol. 44, no. 1, pp. 1–50, 2012.

[45] G. Amati and C. J. Van Rijsbergen Probabilistic models of information retrieval based on measuring the divergence from randomness ACM Transactions on Information Systems (TOIS), vol. 20, no. 4, pp. 357–389, 2002.

[46] C. Macdonald, N. Tonellotto, S. MacAvaney, and I. Ounis Pyterrier: Declarative experimentation in python from bm25 to dense retrieval

Proceedings of the 30th acm international conference on information & knowledge management, 2021, pp. 4526–4533.

[47] S. Robertson, H. Zaragoza et al. The probabilistic relevance framework: Bm25 and beyond Foundations and Trends® in Information Retrieval, vol. 3, no. 4, pp. 333–389, 2009.

[48] J. Ramos et al. Using tf-idf to determine word relevance in document queries Proceedings of the first instructional conference on machine learning, vol. 242, no. 1. Citeseer, 2003, pp. 29–48.

[49] Z. Yang, J. Guo, K. Cai, J. Tang, J. Li, L. Zhang, and Z. Su Understanding retweeting behaviors in social networks Proceedings of the 19th ACM international conference on Information and knowledge management, 2010, pp. 1633–1636.

[50] F. Al-akashi Learning-to-rank: A new web ranking algorithm using artificial neural network International Journal of Hybrid Information Technologies, vol. 1, no. 1, pp. 15–32, 2021.

[51] R. Khoufache, A. Belhadj, H. Azzag, and M. Lebbah Distributed mcmc inference for bayesian non-parametric latent block model arXiv preprint arXiv:2402.01050, 2024.

[52] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong et al. A survey of large language models arXiv preprint arXiv:2303.18223, 2023.

[53] A. Iscen, Y. Avrithis, G. Tolias, T. Furon, and O. Chum Fast spectral ranking for similarity search Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2018, pp. 7632–7641.

[54] A. Malinin and M. Gales Predictive uncertainty estimation via prior networks Advances in neural information processing systems, vol. 31, 2018.

[55] Y. Papakonstantinou and V. Vassalos Query rewriting for semistructured data ACM SIGMOD Record, vol. 28, no. 2, pp. 455–466, 1999.

[56] S. Rizvi, A. Mendelzon, S. Sudarshan, and P. Roy Extending query rewriting techniques for fine-grained access control Proceedings of the 2004 ACM SIGMOD international conference on Management of data, 2004, pp. 551–562.

[57] G. Gottlob, G. Orsi, and A. Pieris Query rewriting and optimization for ontological databases ACM Transactions on Database Systems (TODS), vol. 39, no. 3, pp. 1–46, 2014.

[58] S. Yu, J. Liu, J. Yang, C. Xiong, P. Bennett, J. Gao, and Z. Liu Few-shot generative conversational query rewriting Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval, 2020, pp. 1933–1936.

[59] X. Ma, Y. Gong, P. He, H. Zhao, and N. Duan Query rewriting for retrieval-augmented large language models 2023.

[60] A. R. Rivas, E. L. Iglesias, L. Borrajo et al. Study of query expansion techniques and their application in the biomedical information retrieval The Scientific World Journal, vol. 2014, 2014.

[61] E. N. Efthimiadis Interactive query expansion: A user-based evaluation in a relevance feedback environment Journal of the American Society for Information Science, vol. 51, no. 11, pp. 989–1003, 2000.

[62] E. M. Voorhees, D. K. Harman et al. TREC: Experiment and evaluation in information retrieval. Citeseer, 2005, vol. 63.

[63] M. Sanderson and J. Zobel Information retrieval system evaluation: effort, sensitivity, and reliability Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, 2005, pp. 162–169.

[64] J. Yao, Z. Dou, and J.-R. Wen Employing personal word embeddings for personalized search Proceedings of the 43rd international ACM SIGIR conference on research and development in information retrieval, 2020, pp. 1359–1368.

[65] Abdi, Herve. "Binomial distribution: Binomial and sign tests." Encyclopedia of measurement and statistics 1 (2007).

[66] M. F arber and L. Ao The microsoft academic knowledge graph enhanced: Author name disambiguation, publication classification, and embeddings Quantitative Science Studies, vol. 3, no. 1, pp. 51–98, 2022.

[67] M. Bilal, G. Ali, M. W. Iqbal, M. Anwar, M. S. A. Malik, and R. A. Kadir Auto-prep: efficient and automated data preprocessing pipeline IEEE Access, vol. 10, pp. 107 764–107 784, 2022.

[68] S. Ayesha, M. K. Hanif, and R. Talib Overview and comparative study of dimensionality reduction techniques for high dimensional data Information Fusion, vol. 59, pp. 44–58, 2020.

[69] A. Sun and E.-P. Lim, "Hierarchical text classification and evaluation," in Proceedings 2001 IEEE International Conference on Data Mining. IEEE, 2001, pp. 521–528.

[70] A. K. Uysal and S. Gunal, "The impact of preprocessing on text classification," Information processing & management, vol. 50, no. 1, pp. 104–112, 2014.

[71] K. Kowsari, K. Jafari Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," Information, vol. 10, no. 4, p. 150, 2019.

[72] S. Minaee, N. Kalchbrenner, E. Cambria, N. Nikzad, M. Chenaghlu, and J. Gao, "Deep learning–based text classification: a comprehensive review," ACM computing surveys (CSUR), vol. 54, no. 3, pp. 1–40, 2021.

[73] J. H. Friedman, "Greedy function approximation: a gradient boosting machine," Annals of statistics, pp. 1189–1232, 2001.

[74] L. Alzubaidi, J. Zhang, A. J. Humaidi, A. Al-Dujaili, Y. Duan, O. Al-Shamma, J. Santamar ıa, M. A. Fadhel, M. Al-Amidie, and L. Farhan, "Review of deep learning: concepts, cnn architectures, challenges, applications, future directions," Journal of big Data, vol. 8, pp. 1–74,2021.

[75] A. Sherstinsky, "Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network," Physica D: Nonlinear Phenomena, vol. 404, p. 132306, 2020.

[76] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.

[77] L. Floridi and M. Chiriatti, "Gpt-3: Its nature, scope, limits, and consequences," Minds and Machines, vol. 30, pp. 681–694, 2020.

[78] R. Johnson and T. Zhang, "Deep pyramid convolutional neural networks for text categorization," in Proceedings of the 55th Annual Meet-

ing of the Association for Computational Linguistics (Volume 1: Long Papers), 2017, pp. 562–570.

[79] B. Kuyumcu, C. Aksakalli, and S. Delil, "An automated new approach in fast text classification (fasttext) a case study for turkish text classification without pre-processing," in Proceedings of the 2019 3rd International Conference on Natural Language Processing and Information Retrieval, 2019, pp. 1–4.

[80] S. Abujar, A. K. M. Masum, M. Sanzidul Islam, F. Faisal, and S. A. Hossain, "A bengali text generation approach in context of abstractive text summarization using rnn," Innovations in Computer Science and Engineering: Proceedings of 7th ICICSE, pp. 509–518, 2020.

[81] X. Liu, B. Chai, Y. Wang, and Y. Zhai, "Text classification of enterprise technical requirements based on rcnn att model," in 6th International Conference on Education Reform and Modern Management (ERMM 2021). Atlantis Press, 2021, pp. 513–519.

[82] J. Ding, B. Li, C. Xu, Y. Qiao, and L. Zhang, "Diagnosing crop diseases based on domain-adaptive pre-training bert of electronic medical records," Applied Intelligence, vol. 53, no. 12, pp. 15 979–15 992, 2023.

[83] K. Clark, U. Khandelwal, O. Levy, and C. D. Manning, "What does bert look at? an analysis of bert's attention," arXiv preprint arXiv:1906.04341, 2019.

[84] H. Bao, L. Dong, S. Piao, and F. Wei, "Beit: Bert pre-training of image transformers," arXiv preprint arXiv:2106.08254, 2021.

[85] A. Safaya, M. Abdullatif, and D. Yuret, "Kuisail at semeval-2020 task 12: Bert-cnn for offensive speech identification in social media," arXiv preprint arXiv:2007.13184, 2020.

[86] Y.-J. Li, H.-J. Zhang, W.-M. Pan, R.-J. Feng, and Z.-Y. Zhou, "Microblog rumor detection based on bert-dpcnn," in Artificial Intelligence in China: proceedings of the 2nd international conference on artificial intelligence in China. Springer, 2021, pp. 524–530.

[87] A. Bello, S.-C. Ng, and M.-F. Leung, "A bert framework to sentiment analysis of tweets," Sensors, vol. 23, no. 1, p. 506, 2023.

[88] W.-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, and I. S. Dhillon, "Taming pretrained transformers for extreme multi-label text classification," in Proceedings of the 26th ACM SIGKDD international conference on knowledge discovery & data mining, 2020, pp. 3163–3171.

[89] Xu, J., & Croft, W. B. (1996). *Query expansion using local and global document analysis*. In Proceedings of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 4-11).

[90] Ponte, J. M., & Croft, W. B. (1998). *A language modeling approach to information retrieval*. In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 275-281).

[91] Mori, T., Kokubu, T., & Tanaka, T. (1999). *Cross-lingual information retrieval based on LSI with multiple word spaces*. In Proceedings of the second NTCIR workshop on research in Chinese and Japanese text retrieval and text summarization (pp. 1-8).

[92] Gao, J., Nie, J. Y., Wu, G., & Cao, G. (2001). *Dependence language model for information retrieval*. In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval (pp. 170-177).

[93] Lavrenko, V., & Croft, W. B. (2001). *Relevance based language models*. In Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 120-127).

[94] Finkelstein, L., Gabrilovich, E., Matias, Y., Rivlin, E., Solan, Z., Wolfman, G., & Ruppin, E. (2001). *Placing Search in Context: The Concept Revisited*. ACM Transactions on Information Systems, 20(1), 116-131.

[95] Pagliardini, M., Gupta, P., & Jaggi, M. (2017). Unsupervised learning of sentence embeddings using compositional n-gram features. arXiv preprint arXiv:1703.02507.

[153] Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., & Jaiswal, S. graph2vec: Learning distributed representations of graphs. arXiv 2017. arXiv preprint arXiv:1707.05005.

[96] Lebret, R., & Collobert, R. (2013). Word emdeddings through hellinger pca. arXiv preprint arXiv:1312.5542.

[97] Bengio, Y., Ducharme, R., Vincent, P., & Jauvin, C. (2003). *A Neural Probabilistic Language Model*. Journal of Machine Learning Research, 3, 1137-1155.

[98] Sahlgren, M. (2005). *An Introduction to Random Indexing*. In Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE.

[99] Salakhutdinov, R., Mnih, A., & Hinton, G. (2007). *Restricted Boltzmann machines for collaborative filtering*. In Proceedings of the 24th international conference on Machine learning (pp. 791-798).

[100] Amati, G., & Van Rijsbergen, C. J. (2002). *Probabilistic models of information retrieval based on measuring the divergence from randomness*. ACM Transactions on Information Systems (TOIS), 20(4), 357-389.

[101] Huang, F., & Yates, A. (2009). *Distributional representations for handling sparsity in supervised sequence-labeling*. In Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP (pp. 495-503).

[102] Sutskever, I., Martens, J., & Hinton, G. E. (2011). *Generating Text with Recurrent Neural Networks*. In Proceedings of the 28th International Conference on Machine Learning (ICML-11) (pp. 1017-1024).

[103] Maas, A., Daly, R., Pham, P., Huang, D., Ng, A., & Potts, C. (2011). *Learning Word Vectors for Sentiment Analysis*. In Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (pp. 142-150).

[104] Huang, E., Socher, R., Manning, C., & Ng, A. (2012). *Improving Word Representations via Global Context and Multiple Word Prototypes*. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 873-882).

[105] Klementiev, A., Titov, I., Bhattarai, B. (2012). *Inducing Crosslingual Distributed Representations of Words*. In Proceedings of COLING 2012.

[106] Qiu, X., Tian, L., & Huang, X. (2013). *Latent Semantic Tensor Indexing for Community-based Question Answering*. In Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers) (pp. 434-439).

[107] Kingma, D. P., & Welling, M. (2013). *Auto-Encoding Variational Bayes*. arXiv preprint arXiv:1312.6114.

[108] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013a). *Efficient estimation of word representations in vector space*. In Proceedings of the International Conference on Learning Representations (ICLR).

[109] Mikolov, T., Sutskever, I., Chen, K., Corrado, G. S., & Dean, J. (2013b). *Distributed representations of words and phrases and their compositionality*. In Advances in Neural Information Processing Systems (NIPS) (pp. 3111-3119).

[110] Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2013). *PPDB: The Paraphrase Database*. In Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 758-764).

[111] Bordes, A., Usunier, N., Garcia-Duran, A., Weston, J., & Yakhnenko, O. (2013). *Translating embeddings for modeling multi-relational data*. In Advances in neural information processing systems (pp. 2787–2795).

[112] Le, Q., & Mikolov, T. (2014). *Distributed representations of sentences and documents*. In International conference on machine learning (pp. 1188-1196).

[113] Pennington, J., Socher, R., & Manning, C. (2014). *GloVe: Global Vectors for Word Representation*. In Proceedings of the 2014 Conference on

Empirical Methods in Natural Language Processing (EMNLP), pages 1532–1543, Doha, Qatar. Association for Computational Linguistics.

[114] Tang, D., Wei, F., Yang, N., Zhou, M., Liu, T., & Qin, B. (2014). *Learning Sentiment-Specific Word Embedding for Twitter Sentiment Classification*. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (pp. 1555-1565).

[115] Shen, Y., He, X., Gao, J., Deng, L., & Mesnil, G. (2014b). *Learning semantic representations using convolutional neural networks for web search*. In Proceedings of the 23rd International Conference on World Wide Web (pp. 373-374).

[116] Bahdanau, D., Cho, K., & Bengio, Y. (2014). *Neural Machine Translation by Jointly Learning to Align and Translate*. In Proceedings of the International Conference on Learning Representations (ICLR).

[117] Zheng, G., & Callan, J. (2015). *Learning to reweight terms with distributed representations*. In Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 575-584).

[118] Z. Bian, H. Yang, and J. Tang. (2015) *Knowledge-powered deep learning for word embedding*. In Proceedings of the 24th European Conference on Artificial Intelligence (ECAI), pages 297–304,

[119] Zuccon, G., Koopman, B., Bruza, P., & Azzopardi, L. (2015). *Integrating and evaluating neural word embeddings in information retrieval*. In Proceedings of the 20th Australasian Document Computing Symposium (pp. 1-8).

[120] Sun, Y., Lin, L., Tang, D., Yang, N., Ji, Z., & Wang, X. (2015). *Modeling Mention, Context and Entity with Neural Networks for Entity Disambiguation*. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (pp. 1333-1339)

[121] Ling, W., Luís, T., Marujo, L., Astudillo, R. F., Amir, S., Dyer, C., ... & Trancoso, I. (2015). *Finding function in form: Compositional character models for open vocabulary word representation*. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (pp. 1520-1530).

[122] Arora, S., Li, Y., Liang, Y., Ma, T., & Risteski, A. (2015). *RAND-WALK: A Latent Variable Model Approach to Word Embeddings*. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (pp. 1143-1149).

[123] Wang, H., Wang, N., & Yeung, D. (2015). *Collaborative Deep Learning for Recommender Systems*. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 1235-1244).

[124] Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., & Smith, N. A. (2015). *Retrofitting Word Vectors to Semantic Lexicons*. In Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 1606-1615).

[125] Jiaqiang Chen and Gerard de Melo (2015). *Semantic Information Extraction for Improved Word Embeddings*.

[126] Rush, A. M., Chopra, S., & Weston, J. (2015). *A Neural Attention Model for Abstractive Sentence Summarization*. In Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing (pp. 379–389).

[127] Li, P., Wang, H., Zuo, W., & Zhu, W. (2015). *Text Classification with Topic-based Word Embedding and Convolutional Neural Networks*. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence.

[128] Hashimoto, K., Xiong, C., Tsuruoka, Y., & Socher, R. (2016). *A Joint Many-Task Model: Growing a Neural Network for Multiple NLP Tasks*. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (pp. 1923-1933).

[129] Ma, X., & Hovy, E. (2016). *End-to-end Sequence Labeling via Bi-directional LSTM-CNNs-CRF*. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1064-1074).

[130] Chung, J., Gulcehre, C., Cho, K., & Bengio, Y. (2016). *Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling*. In Proceedings of the 28th International Conference on Neural Information Processing Systems (pp. 207-215).

[131] Kim, Y., Jernite, Y., Sontag, D., & Rush, A. M. (2016). *Character-Aware Neural Language Models*. In Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence (pp. 2741-2749).

[132] Nickel, M., Murphy, K., Tresp, V., & Gabrilovich, E. (2016). *A Review of Relational Machine Learning for Knowledge Graphs*. Proceedings of the IEEE, 104(1), 11-33.

[133] Diaz, F., Mitra, B., & Craswell, N. (2016). *Query expansion with locally-trained word embeddings*. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 367-377).

[134] Camacho-Collados, J., Pilehvar, M. T., & Navigli, R. (2016). *NASARI: Integrating Explicit Knowledge and Corpus Statistics for a Multilingual Representation of Concepts and Entities*. Artificial Intelligence, 240, 36-64.

[135] Wieting, J., Bansal, M., Gimpel, K., & Livescu, K. (2016). *Charagram: Embedding words and sentences via character n-grams*. In Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (pp. 1504-1515)..

[136] Sennrich, R., Haddow, B., & Birch, A. (2016). *Neural machine translation of rare words with subword units*. In Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 1715-1725).

[137] Cheng, J., Wang, Z., & Li, J. (2016). *Set-Based Word Vector Similarity for Document Retrieval*. In Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (pp. 1911-1914).

[138] Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., & Hovy, E. (2016). *Hierarchical Attention Networks for Document Classification*. In Proceedings of the 2016 Conference of the North American Chapter of the Association

for Computational Linguistics: Human Language Technologies (pp. 1480-1489).

[139] Ammar, H., Eaton, E., Ruvolo, P., Taylor, M. (2016). *Unsupervised Cross-Domain Transfer in Policy Gradient Reinforcement Learning*. In Proceedings of the AAAI Conference on Artificial Intelligence 29 (1).

[140] Bhattacharya, P., Goyal, P., & Sarkar, S. (2016). *Using Word Embeddings for Query Translation for Hindi to English Cross Language Information Retrieval*. In Proceedings of the 17th International Conference on Intelligent Text Processing and Computational Linguistics1.

[141] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). *Enriching Word Vectors with Subword Information*. Transactions of the Association for Computational Linguistics, 5, 135-146.

[142] Li, Y., Yu, R., Shahabi, C., & Liu, Y. (2017). *Diffusion Convolutional Recurrent Neural Network: Data-Driven Traffic Forecasting*. arXiv preprint arXiv:1707.01926.

[143] Glavaš, G., & Vulić, I. (2017). *Unsupervised Distributional Hypernymy Detection for High-Dimensional Vectors*. In Proceedings of the 8th International Joint Conference on Natural Language Processing (pp. 235-245).

[144] Bojanowski, P., Grave, E., Joulin, A., & Mikolov, T. (2017). *Enriching word vectors with subword information*. Transactions of the Association for Computational Linguistics, 5, 135-146.

[145] Conneau, A., Kiela, D., Schwenk, H., Barrault, L., Bordes, A. (2017). *Supervised Learning of Universal Sentence Representations from Natural Language Inference Data*. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.

[146] Boag, W., & Kané, H. (2017). *AWE-CM Vectors: Augmenting Word Embeddings with a Clinical Metathesaurus*. arXiv preprint arXiv:1712.01460.

[147] He, X., Liao, L., Zhang, H., Nie, L., Hu, X., & Chua, T. S. (2017). *Neural collaborative filtering*. In Proceedings of the 26th international conference on world wide web (pp. 173-182).

[148] Soliman, A., Eissa, K., & El-Beltagy, S. R. (2017). *Aravec: A set of Arabic word embedding models for use in Arabic NLP*. In Proceedings of the 3rd International Conference on Arabic Computational Linguistics (pp. 1-12).

[149] Dong, Y., Chawla, N. V., & Swami, A. (2017). *metapath2vec: Scalable Representation Learning for Heterogeneous Networks*. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (pp. 135-144)

[150] Garcia, J., Ruiz, F., & Ruiz-Cortes, A. (2017). *Word embeddings for improving REST services discoverability*. In 2017 XLIII Latin American Computer Conference (CLEI) (pp. 1-10). IEEE.

[151] Zamani, H., & Croft, W. B. (2017). *Relevance-based Word Embedding*. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval (pp. 505-514).

[152] Mitra, B., & Craswell, N. (2017). *Neural text embeddings for information retrieval*. In WSDM (Vol. 17, pp. 699-700).

[153] Narayanan, A., Chandramohan, M., Venkatesan, R., Chen, L., Liu, Y., & Jaiswal, S. (2017). *graph2vec: Learning distributed representations of graphs*. arXiv preprint arXiv:1707.05005.

[154] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). *Graph attention networks*. In International Conference on Learning Representations.

[155] McCann, B., Bradbury, J., Xiong, C., & Socher, R. (2017). *Learned in Translation: Contextualized Word Vectors*. In Advances in Neural Information Processing Systems (pp. 2016-2026).

[156] Shwartz, V., Goldberg, Y., & Dagan, I. (2017). *Improving Hypernymy Detection with an Integrated Path-based and Distributional Method*. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. 238-247).

[157] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). *Deep contextualized word representations*. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL-HLT) (Vol. 1, pp. 2227-2237).

[158] Chen, M., Tian, Y., Chang, K., Skiena, S., Zaniolo, C. (2018). *Co-training Embeddings of Knowledge Graphs and Entity Descriptions for Cross-lingual Entity Alignment*. In Proceedings of IJCAI.

[159] Heinzerling, B., & Strube, M. (2018). *BPemb: Fast subword information retrieval in 275 languages*. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation (LREC 2018).

[160] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2018). *Deep Contextualized Word Representations*. In Proceedings of the 2018 Conference of the North American Chapter

of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (pp. 2227-2237).

[161] Mohammad, S., Bravo-Marquez, F., Salameh, M., & Kiritchenko, S. (2018). *SemEval-2018 task 1: Affect in tweets*. In Proceedings of The 12th International Workshop on Semantic Evaluation (pp. 1-17).

[162] Donnat, C., Zitnik, M., Hallac, D., & Leskovec, J. (2018). *Learning Structural Node Embeddings via Diffusion Wavelets*. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining.

[163] Chen, Q., Zhu, X., Ling, Z., Inkpen, D., & Wei, S. (2018). *Neural Natural Language Inference Models Enhanced with External Knowledge*. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers) (pp. XXX-XXX).

[164] Mikolov, T., Grave, E., Bojanowski, P., Puhrsch, C., & Joulin, A. (2018). *Advances in Pre-Training Distributed Word Representations*. In Proceedings of the Eleventh International Conference on Language Resources and Evaluation.

[165] Yin, W., & Schütze, H. (2018). *End-task oriented textual entailment via deep explorations of inter-sentence interactions*. In Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers) (pp. 1694-1704).

[166] Camacho-Collados, J., & Pilehvar, M. T. (2018). *From Word to Sense Embeddings: A Survey on Vector Representations of Meaning*. Journal of Artificial Intelligence Research, 63, 743-788.

[167] Li, J., Yang, Z., Liu, H., & Cai, D. (2018). *Deep rotation equivariant network*. Neurocomputing, 290, 26-33.

[168] Zhang, Y., Qi, P., & Manning, C. D. (2018). *Graph Convolution over Pruned Dependency Trees Improves Relation Extraction*. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (pp. 2205–2215).

[169] Frej, J., Chevallet, J., & Schwab, D. (2018). *Enhancing Translation Language Models with Word Embedding for Information Retrieval*. arXiv preprint arXiv:1801.03844.

[170] Katricheva, N., Yaskevich, A., Lisitsina, A., Zhordaniya, T., Kutuzov, A., & Kuzmenko, E. (2019). *Vec2graph: A Python Library for Visualizing Word Embeddings as Graphs*. In Proceedings of the Thirteenth Workshop on Graph-Based Methods for Natural Language Processing (TextGraphs-13) (pp. 32-41).

[171] Camacho-Collados, J., & Pilehvar, M. T. (2020). *Embeddings in Natural Language Processing: Theory and Advances in Vector Representations of Meaning*. Synthesis Lectures on Human Language Technologies.

[172] Zhang, S., Yao, L., Sun, A., & Tay, Y. (2019). *Deep learning based recommender system: A survey and new perspectives*. ACM Computing Surveys (CSUR), 52(1), 1-38.

[173] Yang, Z., Dai, Z., Yang, Y., Carbonell, J., Salakhutdinov, R., & Le, Q. (2019). *XLNet: Generalized Autoregressive Pretraining for Language Understanding*. arXiv preprint arXiv:1906.08237.

[174] Pires, T., Schlinger, E., Garrette, D. (2019). *How Multilingual is Multilingual BERT?*. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

[175] Devlin, J., Chang, M., Lee, K., & Toutanova, K. (2019). *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers) (pp. 4171-4186).

[176] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., . . . & Stoyanov, V. (2019). *RoBERTa: A robustly optimized BERT pretraining approach*. arXiv preprint arXiv:1907.11692.

[177] Dai, Z., Yang, Z., Yang, Y., Carbonell, J. G., Le, Q. V., & Salakhutdinov, R. (2019). *Transformer-xl: Attentive language models beyond a fixed-length context*. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 2978-2988).

[178] Pilehvar, M. T., & Camacho-Collados, J. (2019). *WiC: 10, 000 Example Pairs for Evaluating Context-Sensitive Meaning Representations*. In Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 1267-1273).

[179] Nogueira, R., & Cho, K. (2019). *Passage re-ranking with BERT*. arXiv preprint arXiv:1901.04085.

[180] Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W., Salakhutdinov, R., & Manning, C. D. (2019). *HotpotQA: A dataset for diverse, explainable*

multi-hop question answering. In Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (pp. 2369-2380).

[181] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2019). *Unsupervised Cross-lingual Representation Learning at Scale*. arXiv preprint arXiv:1911.02116.

[182] Zhang, Z., Han, X., Liu, Z., Jiang, X., Sun, M., & Liu, Q. (2019). *ERNIE: Enhanced Language Representation with Informative Entities*. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (pp. 1441-1451).

[183] Peters, M., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., & Zettlemoyer, L. (2019). *Knowledge Enhanced Contextual Word Representations*. In Proceedings of the Twenty-Fourth International Joint Conference on Artificial Intelligence (pp. XXX-XXX).

[184] Luo, W., Yang, X., Mo, X., Lu, Y., Davis, L. S., Li, J., Yang, J., Lim, S. (2019). *Cross-X Learning for Fine-Grained Visual Categorization*. In Proceedings of the IEEE/CVF International Conference on Computer Vision.

[185] Mi, X., Wang, X., Zhang, Y. (2019). *Cross-Lingual Knowledge Graph Alignment via Graph Matching Neural Network*. In Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics.

[186] Minaee, S., Azimi, E., & Abdolrashidi, A. (2019). *Deep-Sentiment: Sentiment Analysis Using Ensemble of CNN and Bi-LSTM Models*. arXiv preprint arXiv:1904.04206.

[187] Takase, S., Suzuki, J., & Nagata, M. (2019). *Character n-gram Embeddings to Improve RNN Language Models*. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 33, pp. 6842-6849).

[188] Dasgupta, S. S., Boratko, M., Zhang, D., Vilnis, L., Li, X. L., & McCallum, A. (2020). *Improving Local Identifiability in Probabilistic Box Embeddings*. arXiv preprint arXiv:2010.04831.

[189] Alharbi, A. I., & Lee, M. (2020). *Combining character and word embeddings for affect in Arabic informal social media microblogs*. In Natural Language Processing and Information Systems (pp. 287-300). Springer, Cham.

[190] Antoun, W., Baly, F., & Hajj, H. (2020). *AraELECTRA: Pre-training text discriminators for Arabic language understanding*. arXiv preprint arXiv:2004.14196.

[191] Chi, Z., Dong, L., Wei, F., Yang, N., Singhal, S., Wang, W., Song, X., Mao, X., Huang, H., & Zhou, M. (2020). *InfoXLM: An Information-Theoretic Framework for Cross-Lingual Language Model Pre-Training*. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 3576-3588).

[192] Hu, J., Ruder, S., Siddhant, A., Neubig, G., Firat, O., & Johnson, M. (2020). *XTREME: A Massively Multilingual Multi-task Benchmark for Evaluating Cross-lingual Generalization*. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (pp. 3049-3060).

[193] Ryabinin, M., Popov, S., Prokhorenkova, L., & Voita, E. (2020). *Embedding Words in Non-Vector Space with Unsupervised Graph Learning*. In

Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP).

[194] Roy, A., & Pan, S. (2020). *Incorporating Extra Knowledge to Enhance Word Embedding*. In Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (pp. 4929-4935).

[195] Luus, F., Sen, P., Kapanipathi, P., Riegel, R., Makondo, N., Lebese, T., & Gray, A. (2021). *Logic Embeddings for Complex Query Answering*. arXiv preprint arXiv:2103.00418.

[196] Long, Y., Xu, H., Qi, P., Zhang, L., & Li, J. (2021). *Graph Attention Network for Word Embeddings*. In Proceedings of the 20th International Conference on Artificial Intelligence and Security (pp. 215-228).

[197] Dhanith, P. R. J., Surendiran, B., & Raja, S. P. (2023). *A Word Embedding Based Approach for Focused Web Crawling Using the Recurrent Neural Network*. In Proceedings of the International Conference on Advanced Computing and Communication Systems (pp. TBD-TBD).

[198] L. Fang2023, Y. Luo, K. Feng, K. Zhao and A. Hu. (2023). *A Knowledge-Enriched Ensemble Method for Word Embedding and Multi-Sense Embedding*. In IEEE Transactions on Knowledge and Data Engineering, vol. 35, no. 6, pp. 5534-5549, doi: 10.1109/TKDE.2022.3159539.

[199] Li, W., Xue, J., Zhang, X., Chen, H., Chen, Z., Huang, F., & Cai, Y. (2023). *Word-Graph2vec: An Efficient Word Embedding Approach on Word Co-occurrence Graph Using Random Walk Technique*. In Proceedings of the Conference (pp. TBD-TBD).

[200] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., & Harshman, R. (1990). *Indexing by latent semantic analysis*. Journal of the American Society for Information Science, 41(6), 391-407.

[201] Mikolov, T., Chen, K., Corrado, G., & Dean, J. (2013). *Efficient Estimation of Word Representations in Vector Space*. arXiv preprint arXiv:1301.3781.

[202] Landauer, T. K., & Dumais, S. T. (1997). *A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge*. Psychological Review, 104(2), 211–240.

[203] Svedek, T. (2018). *Cross-lingual sentiment analysis using GloVe embeddings*. Master's thesis, University of Ljubljana.

[204] Li, H., Jin, Y., Zheng, W., & Lu, B. (2018). *Cross-Subject Emotion Recognition Using Deep Adaptation Networks*. In Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval.

[205] Hermann, K. M., & Blunsom, P. (2014). *Multilingual Models for Compositional Distributed Semantics*. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers).

[206] Rücklé, A., & Gurevych, I. (2017). *Representation Learning for Answer Selection with LSTM-Based Importance Weighting*. In Proceedings of the 12th International Conference on Computational Semantics (IWCS 2017).

[207] Hamilton, W., Ying, Z., & Leskovec, J. (2017). *Inductive representation learning on large graphs*. In Advances in Neural Information Processing Systems (pp. 1024-1034).

[208] Grover, A., & Leskovec, J. (2016). *node2vec: Scalable feature learning for networks*. In Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining (pp. 855-864).

[209] Bai, X., Bian, J., & Wang, B. (2018). *Graph2vec: Learning distributed representations of graphs*. In Proceedings of the 27th ACM International Conference on Information and Knowledge Management (pp. 2301-2304).

[210] Abu-El-Haija, S., Perozzi, B., Al-Rfou, R., & Alemi, A. (2019). *Towards generalizable node embeddings*. In Proceedings of the 28th ACM International Conference on Information and Knowledge Management (pp. 1809-1812).

[211] Veličković, P., Cucurull, G., Casanova, A., Romero, A., Liò, P., & Bengio, Y. (2018). *Graph attention networks*. In 6th International Conference on Learning Representations (ICLR 2018).

[212] Kipf, T. N., & Welling, M. (2017). *Semi-supervised classification with graph convolutional networks*. In 5th International Conference on Learning Representations (ICLR 2017).

[213] Zheng, G., Callan, J., 2015. *Learning to reweight terms with distributed representations.* In: Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval. ACM, pp. 575–584

[214] Lebret, R., & Collobert, R. (2014). *Word embeddings through Hellinger PCA*. In Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (pp. 482-490).

[215] Purpura, A., Maggipinto, M., Silvello, G., & Susto, G. A. (2019). *Probabilistic Word Embeddings in NIR: A Promising Model That Does Not Work as Expected (For Now)*. In The 2019 ACM SIGIR International Conference on the Theory of Information Retrieval (ICTIR '19), October 2–5, 2019, Santa Clara, CA, USA. ACM, New York, NY, USA.

[216] Mnih, A., & Hinton, G. (2008). *A Scalable Hierarchical Distributed Language Model*. In Proceedings of the Neural Information Processing Systems (pp. 1081-1088)

[217] Mori, T., Kokubu, T., & Tanaka, T. (2001). *Cross-Lingual Information Retrieval based on LSI with Multiple Word Spaces*. In Proceedings of the Second NTCIR Workshop on Research in Japanese Text Retrieval and Term Recognition (pp. 1-8)

[218] AuthorLastName, AuthorFirstName. (Year). *Constraining Word Embeddings by Prior Knowledge – Application to Medical Information Retrieval*. In Proceedings of the Conference (pp. page numbers).

[219] Kim, Y. (2014). *Convolutional neural networks for sentence classification*. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP).

[220] Zhang, X., Zhao, J., & LeCun, Y. (2015). *Character-level convolutional networks for text classification*. Advances in neural information processing systems.

[221] Lai, S., Xu, L., Liu, K., & Zhao, J. (2015). *Recurrent convolutional neural networks for text classification*. Twenty-ninth AAAI conference on artificial intelligence.

LIST OF PUBLICATIONS

[1] **Lovely Yeswanth P, Satya Krishna N, Lavanya P, Sriya P, Yogeshvar Reddy K** (2024) Neural Information Retrieval: Word Embedding Literature Review, *International Journal of Multimedia Information Retrieval (IJMIR)* Under review

[2] **Lovely Yeswanth P, Satya Krishna N, Lavanya P, Sriya P, Yogeshvar Reddy K**, (2024) Modified Cross Encoder for Two Stage Passage Ranking, *International Symposium on Innovative Approaches in Smart Technologies (ISAS2024)*, Under review

[3] **Lovely Yeswanth P, Satya Krishna N, Lavanya P, Sriya P**, (2024) Improving Information Retrieval Task with Asymmetric Divergence from Randomness in Query Expansion, *Future Technologies Conference (FTC)*, Under review

[4] **Lovely Yeswanth P, Satya Krishna N, Lavanya P, Sriya P**, (2024) Efficient English text classification using Bert-DPRCNN, *International Journal of Multimedia Information Retrieval (FTC)*, Under review