Improving the robustness and effectiveness of neural retrievers in noisy and low-resource settings

Sidiropoulos, G.

Publication date
2025

Document Version
Final published version

IMPROVING THE ROBUSTNESS AND EFFECTIVENESS OF NEURAL RETRIEVERS IN NOISY AND LOW-RESOURCE SETTINGS

Georgios Sidiropoulos

Improving the Robustness and Effectiveness of Neural Retrievers in Noisy and Low-Resource Settings

Georgios Sidiropoulos

Improving the Robustness and Effectiveness of Neural Retrievers in Noisy and Low-Resource Settings

ACADEMISCH PROEFSCHRIFT

ter verkrijging van de graad van doctor aan de
Universiteit van Amsterdam
op gezag van de Rector Magnificus
prof. dr. ir. P.P.C.C. Verbeek
ten overstaan van een door het College voor Promoties ingestelde
commissie, in het openbaar te verdedigen in
de Agnietenkapel
op dinsdag 7 januari 2025, te 16.00 uur

door

Georgios Sidiropoulos

geboren te Athina

Promotiecommissie

Promotor:	prof. dr. E. Kanoulas	Universiteit van Amsterdam
Co-promotor:	prof. dr. M. de Rijke	Universiteit van Amsterdam
Overige leden:	prof. dr. S.I. Birbil	Universiteit van Amsterdam
	prof. dr. J.A. Good	Universiteit van Amsterdam
	dr. L. Màrquez	Amazon
	dr. S. Rudinac	Universiteit van Amsterdam
	prof. dr. G. Zuccon	University of Queensland

Faculteit Economie en Bedrijfskunde

The research was carried out at the Information Retrieval Lab and the Business Analytics section at the University of Amsterdam, with support from H2020-EU.3.4. - SOCIETAL CHALLENGES - Smart, Green And Integrated Transport under project number 814961.

Printed by Proefschriftspecialist, Zaandam, The Netherlands

ISBN: 978-94-93391-76-5

Acknowledgements

First and foremost, I would like to express my deepest gratitude to my supervisors. Evangelos, thank you for giving me the opportunity to pursue a PhD, for your trust in my interests, and for allowing me to explore topics that inspired me. Your guidance and understanding have been invaluable throughout this journey. Maarten, thank you for your support and for welcoming me into ILPS. Your leadership and consistency have been admirable. Niko, although you were not officially my supervisor, your support has been constant since my MSc days. I am deeply grateful for all the advice, encouragement, and mentorship you have provided along the way.

Next, I would like to thank Guido, Ilker, Judith, Lluís, and Stevan for agreeing to serve as committee members and reviewing my thesis.

Furthermore, I want to thank my colleagues from the IRLab and Business Analytics section for sharing their inspiring ideas in countless discussions, one-on-one meetings, and presentations—special thanks to Antonis and Sam for being more than colleagues.

For the last eight years, I have been living in Amsterdam, and a significant part of this journey has been the friendships I have formed here. Thanos R., Thanos E., Harris, and Deb, thank you for the games, the fun times, and the meaningful conversations. Sharing the ups and downs of expatriate life with you has made the experience much richer. Stelio, thank you for your friendship and always standing by your principles.

Finally, I want to express my heartfelt thanks to my nearest and dearest. To my lifelong friends Antonis and Yannis, thank you for your unwavering friendship. Juci, thank you for making me look forward to life after the PhD. Father, thank you for your support and for setting an example of hard work and discipline. Brother, thank you for always being there to talk, and for your understanding and constant care. Malorie, thank you for the fun times, your caring, and for standing by my side during the most challenging times of this journey; your support has been invaluable.

Mana, this thesis is dedicated to you.

Georgios Sidiropoulos
Diemen, October 2024

Contents

1 Introduction

The primary goal of a search engine is to facilitate user navigation through the immense and diverse amount of information available in online repositories and ensure that users can successfully find the information they seek [130]. In order to fulfill the user's information needs, search engines need to efficiently and effectively locate information that is meant to be relevant to the user (in terms of satisfying the user's information need) and then present it to them. Millions of users interact with various search engines daily for their information needs [13, 69]. The process begins with the user specifying their information need through a query, which triggers a search for information resources that are likely to be relevant to the user's request. The retrieved information is then presented to the user through the search engine result pages.

One of the main tasks performed by a search engine, and a classic information retrieval (IR) problem, is ad-hoc retrieval [108]. Given a query and a document collection, the search engine returns a ranked list of documents so that the most relevant documents are ranked higher compared to the less relevant ones [129]. A query typically contains a handful of terms, while documents may range from a few terms to passages of text to an entire book, depending on the underlying collection of textual data. Over the past few years, the IR community has put extensive efforts into developing neural retrievers for ad-hoc retrieval [108]. These retrievers employ shallow or deep neural networks to effectively retrieve relevant search results in response to a given query. Neural retrievers have brought significant improvements and have managed to alleviate several limitations of the traditional retrieval approaches [73, 82, 108, 127]. Some of the traditional approaches to ad-hoc retrieval are presented below alongside their drawbacks.

Traditional approaches to ad-hoc retrieval, such as BM25 [128] and TF-IDF [132], are *lexical-based* and rely on exact term matching between the query and the documents. Due to their effectiveness and high efficiency, such approaches have become widely adopted in search engines and remain competitive against more advanced techniques until today. However, there is often a discrepancy between the terms used in a user's query and the terms present in relevant documents [41]. A major limitation of these lexical-based approaches is that relevance is computed based on the frequency of the query terms occurring in the document and, therefore, that they cannot capture semantic relationships between queries and documents [87].

To bridge the semantic gap, other classic retrieval approaches used *topic modeling*

methods (e.g., based on LDA [170] and pLSA [62]) for document representation by a finite number of latent topics [4, 170]. Unlike lexical-based approaches, the relevance to a query of a document is not determined by the frequency of the query terms in the document but rather by the frequency of query terms in the latent topics and the likelihood that those respective topics generate the document under consideration. However, topic models for retrieval are not explicitly trained to learn to rank documents but instead are trained to capture the underlying thematic structure within a collection of documents. Consequently, many studies have found topic models to be ineffective for ad-hoc retrieval [see, e.g., 3, 180].

On the other end of the spectrum, learning-to-rank (LTR) trains non-neural and neural models with ranking objectives over features that are usually hand-crafted [14, 93]. In this framework, each query-document pair is encoded into a feature vector, and the (ranking) model is trained to map the vector to a real-valued score such that a rank-based metric is maximized. Therefore, the neural network is used only for query-document matching. LTR approaches may rely on the assumption that certain features, which cannot be obtained solely from the query and document content, are available. An example is the trustworthiness of a website. These manually engineered features are domain-specific and do not necessarily generalize well to new domains [16, 48]. Additionally, defining, extracting, and validating these features can be time-consuming [155]. An important shortcoming of LTR is that it cannot scale to millions of documents due to the computational cost associated with computing the features and that standard document indexing structures do not always support these features [31]. For this reason, LTR is usually employed in a multi-stage retrieval manner to re-rank small sets of documents retrieved by another (more efficient) system, e.g., BM25 [128].

Neural retrieval [82, 107, 136] addresses many of the shortcomings of the aforementioned methods. In neural retrieval the models do not rely on hand-crafted features; instead, they take the raw text[1] of the query and document as input and learn representations for that text (i.e., embeddings). The models can be divided into two main categories:

- First-stage retrievers: A key aspect of first-stage neural retrievers (for simplicity, we will refer to them as neural retrievers) is that they learn separate latent representations for a query and a document. Rather than using the neural network for matching, neural retrievers use it to learn effective, separate latent representations for a query and a document, bridging the vocabulary gap between the two. At the same time, they employ a simple similarity metric for matching (e.g., cosine or dot product). One critical attribute of neural retrievers that allows for their deployment to real-world applications, which requires searching over millions of documents, is their indexability. Documents can be encoded and indexed offline, while at query time, high-scoring documents with respect to a query can be found using efficient maximum inner product search (MIPS) [70]. Figure 1.1a depicts the architecture of neural retrievers and how retrieval works at query time.

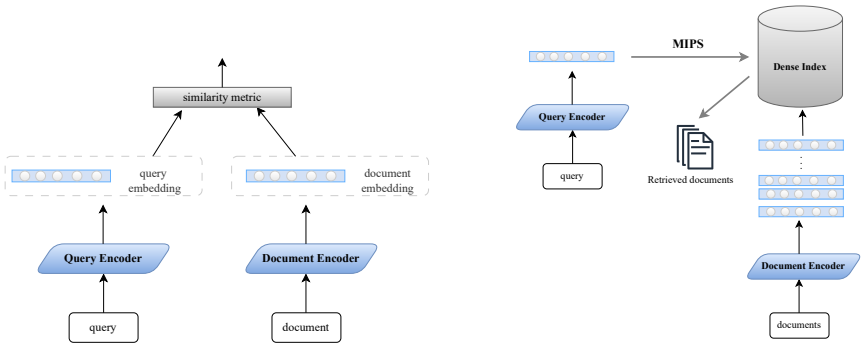
[1] We refer to textual queries and documents without loss of generality. Neural retrievers are not limited to text; they can support various modalities such as image and audio. Additionally, they can facilitate retrieval between the same modalities (e.g., text query and document) and different modalities (e.g., spoken query and text document).

- Re-rankers: Neural re-rankers concatenate the query with the document and pass it as input to a neural network that outputs a relevance score. Such models can typically achieve higher performance than neural retrievers [94]. However, due to their architecture, they are inherently non-indexable. At retrieval time, given a query, all the query-document pairs need to pass through the neural re-ranker to compute their relevance. Due to the prohibitive retrieval-time latency, neural re-rankers are used only for ranking small sets of documents [115].

In this thesis, we focus on (first-stage) neural retrievers. The effective and efficient training of neural retrievers, as well as the indexing and searching of dense numerical vectors has been enabled due to the availability of (i) large-scale labeled datasets (MS-MARCO [113], NaturalQuestions [78], HotpotQA [179] etc.), (ii) large pre-trained language models for semantic matching (e.g., BERT [32]), (iii) powerful computational resources that have made training large-scale neural models feasible, and (iv) better techniques to support indexing and efficient MIPS (e.g., FAISS [70]). Recently, the field of neural retrieval witnessed a revolutionary innovation with the introduction of pre-trained language models and training setups (i.e., loss functions such as InfoNCE [116] and learning paradigms including batch contrastive learning and knowledge distillation) that allow neural retrievers to learn effective query and document encoders only by fine-tuning on query-document pairs (i.e., no need to pre-train the retriever with an inverse cloze task) [73]. Neural retrievers use pre-trained language models as encoders and employ training paradigms such as supervised batch contrastive learning, forcing the model to increase its predicted similarity between the relevant query-document pairs and decrease its predicted similarity of the irrelevant ones. Such retrievers are commonly known as dense retrievers. As a result of their high performance [73, 127, 188], dense retrievers have become the new paradigm in ad-hoc retrieval.

At this point it is important to discuss some of the desiderata of search engines. Search engines must be able to deal with user queries that are common, unpopular, or have never been seen before, and with queries that are error-free or contain errors, e.g., typographical errors in textual queries and background noise in spoken queries. Queries with errors, similar to unpopular queries that have not been encountered before, are generally not included in the data used to train the underlying retrieval model. Therefore, when such queries are encountered during inference, they produce out-of-distribution inputs for the retriever. Furthermore, search engines must be able to adapt to new domains (e.g., emerging markets). To this extent, the underlying retriever must handle out-of-domain queries, e.g., queries on topics, genres, styles, or languages different from the ones used during training. Even though neural retrievers have shown great effectiveness in in-distribution and in-domain queries, recent studies have unveiled that they do not consistently demonstrate such robustness characteristics "out of the box" when it comes to out-of-distribution and out-of-domain queries [154, 196].

In addition, search engines need to operate effectively in resource-constrained environments to ensure accessibility and scalability [17, 100, 152]. Training and inference of effective neural retrievers also come at a cost in terms of ever-growing computational and data resources required for training these increasingly complex models [45]. Obtaining large-scale supervised datasets for training can have a significant financial cost since it requires human annotation [86, 145, 146]. In addition, computational



(a) Neural retriever architecture.

(b) Index and search for neural retrieval.

Figure 1.1: In neural retrieval, neural encoders are used to embed a query and a document into separate dense vectors. A simple similarity metric is computed over the two vectors to estimate the relevance between the query and document (left figure). All the available documents are encoded into an index of dense vectors offline, and the retrieval with respect to a query is implemented with efficient MIPS over the index (right figure).

resources require substantial financial investment for their acquisition, operation, and maintenance [88, 148]. The training and inference of neural models also have significant energy costs and carbon footprints [134]. Therefore, data and computational efficiency during training and inference are essential [111, 131]. Hence, the desired properties of the underlying retrieval system include robustness and effectiveness in out-of-domain, out-of-distribution, and low-resource settings.

That said, in order to fulfill the user's expectations and address the increasing complexity of their queries, modern search engines need to (i) provide access to unstructured knowledge sources, e.g., textual documents, social media posts, and images as well as structured knowledge sources, e.g., knowledge graphs containing real-world entities-such as objects, events, situations or concepts—and the relationships between them, (ii) further provide question answering functionalities to enhance the search engine result page with direct answers to the given queries, and finally (iii) support multiple means via which the users can express their queries e.g., text, voice, and image.

In this thesis, we focus on developing robust neural retrievers to support various functionalities of modern search engines. In particular, we investigate *how to improve the robustness and effectiveness of neural retrievers in noisy and low-resource settings*. We first explore the impact that errors in a query have on the retrieval performance of neural retrievers for ad-hoc retrieval and propose ways to robustify them. Then, we explore the use of neural methods for multi-modal retrieval over spoken queries and textual documents. Next, we examine the challenges of training an effective neural retriever with limited computational resources to tackle complex user queries requiring multi-hop retrieval. Lastly, we investigate how neural retrieval can be used to increase relation prediction performance in KGQA over previously unseen domains.

1.1 Research Outline and Questions

We scope the thesis around four research themes aimed at improving the robustness and effectiveness of neural retrievers in noisy and low-resource settings for their application in search engines: (i) improving the robustness of neural retrievers against typos (Chapters 2, 3, and 4), (ii) improving the robustness of speech-based search with neural retrieval (Chapters 5 and 6), (iii) improving the efficiency of training neural retrievers on low resources for multi-hop retrieval (Chapter 7), and (iv) improving domain adaptation in KGQA with neural retrieval (Chapter 8).

1.1.1 Improving the robustness of neural retrievers against typos

It is of vital importance that a search engine does not assume error-free queries. Users can introduce typographical errors, commonly known as typos, while submitting queries to a search engine. These typos may result from keyboard errors caused by the spatial proximity of the keys or misplaced fingers, phonetic typing errors due to close pronunciation of words, and misspellings. Hence, the underlying retrieval system must be robust against queries with typos. Even though dense retrievers are highly effective on typo-free queries, preliminary studies have shown a dramatic drop in retrieval performance when dealing with typoed queries. We aim to answer the following research question:

RQ1 Can we robustify dense retrievers against queries with typos?

We propose an alternative training setup for the dense retriever, which aims to learn better representations from noisy text. Our approach combines data augmentation with contrastive learning, aiming to maximize the agreement between differently augmented views of the same object, namely, the original typo-free query and its typoed variation. Our results show that our approach not only improves robustness against typos but also performs better than separately applying data augmentation or contrastive learning.

Most work on robustifying dense retrievers relies on data augmentation during training to generate synthetic typoed queries and additional robustifying subtasks to align the original, typo-free query with its typoed variant. While multiple typoed variants exist as positive samples per query, most existing methods rely on a single positive sample and a set of negatives per anchor and employ contrastive learning to tackle the robustifying subtask, thereby limiting the usage of the multiple positives. An interesting question that arises is:

RQ2 Can we improve the robustness of dense retrievers with contrastive learning in a way that accounts for multiple positives and negatives?

We first set to identify cases in typo-robust dense retrieval where the robustifying subtasks take into consideration only a single positive sample and optimize a contrastive loss, even though multiple ones are available. Next, we replace the contrastive loss with its multi-positive alternative, which uses all available positives. Our results demonstrate that employing multi-positive contrastive learning yields improvements in robustness compared to contrastive learning with a single positive.

While learning better representations for noisy text is one possible direction for improving robustness against typos, it is not the only one. Uncertainty estimation

can also be considered as a viable option. Uncertainty-aware retrieval systems tend to be more robust to noise and variability in the input data [24]. Furthermore, when uncertainty estimates are provided alongside retrieval results, the transparency and interpretability of the system's decision-making process are improved. A limitation of current approaches in dense retrieval is that learned representations cannot capture or express uncertainty. The current paradigm in dense retrieval is to represent queries and documents as low-dimensional real-valued vectors using pre-trained language models as encoders and then compute query-document similarities as the dot product of their respective vector representations. The multivariate representation learning (MRL) framework proposed by Zamani and Bendersky [187] is the first method that works in the direction of modeling uncertainty in dense retrieval. This framework represents queries and documents as multivariate normal distributions rather than vectors and computes query-document similarity as the negative KL divergence between these distributions.

RQ3 Can MRL capture uncertainty in queries that contain typos?

Inspired by work in computer vision, where uncertainty-aware models assign higher uncertainty to corrupted data and lower to clean [168], we argue that the MRL framework should assign higher variance to the typoed queries than to typo-free ones. To answer this research question, our study begins with the reproduction of the MRL framework since neither the source code nor model checkpoints are released. We attempt to reproduce MRL under memory constraints (e.g., an academic computational budget). In particular, we focus on a memory-limited, single GPU setup. Even though we successfully trained an effective dense retriever, we could not reproduce the results reported in the original paper or uncover the reported trends against the baselines under a memory-limited setup – that facilitates fair comparisons of MRL against its baselines. We believe that the impressive results reported in the original paper are due to training with a large batch size for many training steps. We unveil that the MRL cannot consistently capture uncertainty since it assigns lower uncertainty to typoed queries than typo-free ones.

1.1.2 Improving the robustness of speech-based search with neural retrieval

Multimodality has become increasingly important in modern search engines [35, 135, 189]. It enhances the user experience by creating a more immersive and engaging environment and can further improve accessibility and inclusivity by accommodating users with different needs. Nowadays, millions of users interact with search engines via speech interfaces [137]. Voice interaction has become increasingly popular due to its convenience, hands-free operation, natural user experience, and support for users with visual and motor impairments for whom conventional text entry mechanisms (i.e., keyboards) are not applicable [121]. In our next study, we focus on speech-based search, where queries are in spoken form and the documents are in textual form.

A straightforward approach that does not necessarily require retraining a retrieval system to tackle spoken queries combines an automatic speech recognition (ASR) system with a text retriever. In ASR-Retriever pipeline approaches, the spoken queries are transcribed with an ASR model and then passed through the text retriever. The

ASR system cannot always guarantee a perfect transcription, e.g., difficult accents, background noise, and rare entities can lead to a corrupted transcription.

RQ4 Are dense retrievers robust against queries that contain transcription errors?

We showcase that state-of-the-art dense retrievers are not inherently robust against queries with ASR errors. However, dense retrievers trained to be robust against typographical errors also show robustness against ASR noise. That said, we propose training with transcriptions from synthetically generated spoken queries and find that it yields the best improvements in robustness.

It is important to note that in an ASR-Retriever pipeline, ASR propagates its errors to the downstream retriever. As a result, a higher word error rate from the ASR system can negatively affect the overall performance of the pipeline. Additionally, training an ASR model requires obtaining a large amount of annotated speech. This can negatively impact the applicability of ASR-Retriever pipelines for low-resource language scenarios. Therefore, bypassing the ASR system could alleviate some of the aforementioned drawbacks. This is the focus of our next research question.

RQ5 How does a multimodal dense retriever perform in speech-based search?

We propose a multimodal dense retriever that does not require an ASR model and can be trained end-to-end. Our method adapts the standard dense retriever architecture, wherein the pre-trained language models are used as query and document encoders (i.e., where query and document are in text form) by replacing the backbone language model used to encode the queries with a self-supervised speech model. Our findings suggest that, on shorter questions, our multimodal retriever is a promising alternative to the ASR-Retriever pipeline. It obtains better retrieval performance than ASR-Retriever pipelines in cases where the ASR system tends to mistranscribe crucial words in the question or produce transcriptions with a high word error rate.

1.1.3 Improving the efficiency of training neural retrievers on low resources for multi-hop retrieval

The advent of advanced search engines has been accompanied by the evolution of users' expectations, leading to more complex and nuanced information needs [1]. As users express their information needs in complex queries, modern search engines must be able to deal with such queries. Complex queries cannot be resolved with a single document; they require multiple documents to provide adequate evidence collectively. Thus, complex queries need multi-hop retrieval that involves iteratively retrieving documents and reformulating the query in order to retrieve different documents at each hop [80]. For instance, "Where was Barack Obama born?" and "Who was the defense counsel of a German woman who underwent Catholic exorcism rites during the year before her death?" are two examples of simple and complex queries, respectively. Dense retrievers have achieved state-of-the-art performance on multi-hop retrieval for complex queries [175]. However, multi-hop dense retrievers are computationally intensive, requiring multiple GPUs to train an effective retriever. Their applicability to low-resource scenarios has not been properly studied. We aim to answer this:

RQ6 Can we train an effective dense retriever for multi-hop retrieval with limited computing resources?

Instead of training an end-to-end multi-hop dense retriever, we propose a hybrid retrieval approach, combining lexical and dense retrievers. Our results show that when trained with limited resources, end-to-end multi-hop dense retrievers witness a dramatic drop in performance, while our hybrid approach outperforms them. Further, our hybrid approach remains highly competitive even against multi-hop dense retrievers trained on significantly more resources.

1.1.4 Improving domain adaptation in KGQA with neural retrieval

Modern search engines are enhanced with question answering (QA) functionalities to provide direct answers to users' queries; in particular, the search engine result page is enhanced with an answer box [77, 186]. Many queries refer to real-world entities, and therefore, QA over knowledge graphs can be used to answer them. Due to the dynamic nature of real-world information, KGs are also dynamic and designed to adapt to new entities and relationships. As a result, QA systems that provide access to KGs should be able to deal with queries that refer to previously unseen domains. Our preliminary experiments show a dramatic drop in the KGQA systems' performance over queries from previously unseen domains. Our findings further indicate that this decrease originates from relation prediction. Therefore, we set out to explore how to improve the robustness of the relation prediction model for new domains:

RQ7 Can neural retrieval combined with data augmentation increase the relation prediction robustness of a KGQA system over previously unseen domains?

In contrast to most work that treats relation prediction as a classification task, thereby limiting the applicability to new domains with new entities and entity relations, we follow a retrieval approach by using the textual label to represent the relation. This way we can, in principle, represent any relation during inference time and hence, account for new domains. To this end, we employ a neural retriever. To further increase its robustness, we augment the training set with synthetically generated queries for the new domains. Our results show that we can robustify relation prediction, and the overall KGQA system, and outperform model-based approaches.

1.2 Main Contributions

In this section, an overview of the key contributions of this thesis is presented.

In this thesis we make the following algorithmic contributions:

1. A method that combines data augmentation with contrastive learning for improving the robustness of neural retrievers (Chapter 2).

2. A method that uses contrastive learning with multiple positives and negatives for improving the robustness of neural retrievers (Chapter 3).

3. A multimodal neural retriever for speech-based search (Chapter 6).
4. A hybrid method, combining lexical and neural models, for multi-hop retrieval over complex queries (Chapter 7).
5. A framework for increasing the domain adaptation capabilities of KGQA systems, based on robustifying the relation prediction with neural retrieval (Chapter 8).

We make the following empirical contributions:

6. An empirical comparison of a proposed typo-robust neural retrieval model vs. other baselines, and an analysis of how different types of typos affect the retrieval performance (Chapter 2).
7. An empirical comparison of a proposed method that adopts a multi-positive contrastive loss vs. other baselines with single positive and multiple negatives, and an analysis of the effectiveness of the former w.r.t. the number of positives (Chapter 3).
8. A reproducibility study of the multivariate representation learning framework for dense retrieval alongside an open-source implementation. A fair comparison of the proposed framework against its competitors and an extensive ablation study. An additional experimental setting for testing the model's ability to capture uncertainty on typoed queries (Chapter 4).
9. An empirical comparison of a neural retriever trained with data augmentation on synthetic speech and other typo-robust neural baselines, and an extensive analysis of the robustness over different accents, varying amounts of synthetic data available during training, and synthetic vs. natural speech (Chapter 5).
10. An empirical comparison of an end-to-end trained multimodal neural retriever vs. ASR-Retriever pipeline baselines. An analysis of the impact of ASR mistranscription of important words on ASR-Retriever pipelines, and an ablation study concerning different training schemes (Chapter 6).
11. An empirical comparison of a hybrid retriever that combines lexical with dense retrieval, against pure dense and pure lexical counterparts, and an analysis of the impact that training on low resources has on the different methods (Chapter 7).
12. An empirical comparison of ways to generate synthetic queries for the new domains. An empirical comparison of a data-centric domain adaptation approach against a model-centric, and an empirical comparison of a proposed KGQA model against state-of-the-art KGQA systems for the in-domain setting (Chapter 8).

The thesis contributes the following resources:

13. Challenging test sets for testing robustness against typos in ad-hoc retrieval; queries contain synthetically created realistic typos on non-stop words and highly discriminative utterances (Chapter 2).

14. A large-scale dataset for speech-based search that contains approximately 400K synthetically generated spoken queries (Chapter 5).

15. A distant-supervision-based tool for extracting keywords for entity relationships in knowledge graphs (Chapter 8).

1.3 Thesis Overview

This thesis is organized into four parts, all of which focus on building robust neural retrieval models to support the various functionalities of modern search engines. In the first part, we study how to robustify dense retrievers against typos in the query for the task of ad-hoc retrieval. In particular, in Chapter 2 and Chapter 3, we propose methods for robustifying dense retrievers by learning better representations for the noisy text, while in Chapter 4, we explore a dense retrieval model that can model uncertainty. In the second part, we study how to support multimodal retrieval with neural retrievers. In Chapter 5, we focus on the ASR-Retriever pipeline and try to robustify textual dense retrievers against ASR noise. In contrast, in Chapter 6, we explore an end-to-end multimodal dense retrieval approach. In the third part, Chapter 7, we study how to build neural retrievers to support complex queries that require multi-hop retrieval when limited computational resources are available. In the fourth part, we study how to robustify KGQA systems on previously unseen domains with neural retrieval and synthetic query generation. Finally, we conclude the thesis and discuss directions for future work in Chapter 9.

1.4 Origins

Below, we list the publications that serve as the primary sources for each chapter.

Chapter 2 is based on the paper:

> G. Sidiropoulos and E. Kanoulas. Analysing the robustness of dual encoders for dense retrieval against misspellings. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2022 [139].
>
> GS: Conceptualization, Investigation, Methodology, Data curation, Software, Writing – original draft. EK: Conceptualization, Supervision, Funding acquisition, Writing – review & editing.

Chapter 3 is based on the paper:

> G. Sidiropoulos and E. Kanoulas. Improving the robustness of dense retrievers against typos via multi-positive contrastive learning. In *Advances in Information Retrieval - 46th European Conference on Information Retrieval, ECIR, Proceedings, Part III*. Springer, 2024 [140].

GS: Conceptualization, Investigation, Methodology, Software, Writing – original draft. EK: Conceptualization, Supervision, Funding acquisition, Writing – review & editing.

Chapter 4 is based on the paper:

G. Sidiropoulos, S. Bhargav, P. Eustratiadis, and E. Kanoulas. Multivariate dense retrieval: A reproducibility study under a memory-limited setup. *Under submission* [141].

GS: Conceptualization, Investigation, Validation, Software, Writing – original draft. SB: Conceptualization, Investigation, Validation, Software, Writing – original draft. PE: Conceptualization, Supervision, Writing – review & editing. EK: Conceptualization, Supervision, Funding acquisition, Writing – review & editing.

Chapter 5 is based on the paper:

G. Sidiropoulos, S. Vakulenko, and E. Kanoulas. On the impact of speech recognition errors in passage retrieval for spoken question answering. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, 2022 [144].

GS: Conceptualization, Investigation, Methodology, Data curation, Software, Writing – original draft. SV: Conceptualization. EK: Conceptualization, Supervision, Funding acquisition, Writing – review & editing.

Chapter 6 is based on the paper:

G. Sidiropoulos and E. Kanoulas. Multimodal dense passage retrieval for open-domain spoken question answering. *Under submission* [138].

GS: Conceptualization, Investigation, Methodology, Software, Writing – original draft. EK: Conceptualization, Supervision, Funding acquisition, Writing – review & editing.

Chapter 7 is based on the paper:

G. Sidiropoulos, N. Voskarides, S. Vakulenko, and E. Kanoulas. Combining lexical and dense retrieval for computationally efficient multi-hop question answering. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing, SustaiNLP@EMNLP 2021*. Association for Computational Linguistics, 2021 [143].

GS: Conceptualization, Investigation, Methodology, Software, Writing – original draft. NV: Conceptualization, Supervision, Writing – review & editing. SV: Conceptualization, Supervision, Methodology, Software, Writing – review & editing. EK: Conceptualization, Supervision, Funding acquisition, Writing – review & editing.

Chapter 8 is based on the paper:

> G. Sidiropoulos, N. Voskarides, and E. Kanoulas. Knowledge graph simple question answering for unseen domains. In *Conference on Automated Knowledge Base Construction, AKBC*, 2020 [142].
>
> GS: Conceptualization, Investigation, Methodology, Software, Writing – original draft. NV: Conceptualization, Supervision, Writing – review & editing. EK: Conceptualization, Supervision, Funding acquisition, Writing – review & editing.

The present thesis has benefited indirectly from insights gained through the following publications:

- S. Bhargav, G. Sidiropoulos, and E. Kanoulas. 'It's on the tip of my tongue': A new dataset for known-item retrieval. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining*. ACM, 2022 [8].
- P. Jonk, V. de Vries, R. Wever, G. Sidiropoulos, and E. Kanoulas. Natural language processing of aviation occurrence reports for safety management. In *Proceedings of the 32nd European Safety and Reliability Conference*. RPS, 2022 [71].

Part I

Improving the Robustness of Neural Retrievers Against Typos

2

Data Augmentation and Contrastive Learning for Typo-robust Dense Retrieval

Search engines can not assume error-free queries. Common types of error that users introduce to queries are typographical errors, often called typos, due to the spatial proximity of the keys on a keyboard, phonetic typing errors due to close pronunciation of words, and misspellings. As a result, the underlying retrieval system must be robust against potential errors in the query. In the first part of the thesis, we focus on improving the robustness of dense retrievers against typos. Even though dense retrievers achieve high performance when dealing with typo-free queries, preliminary studies have shown a dramatic drop in retrieval performance on queries that contain typos. In this first chapter, we set to answer **RQ1**: Can we robustify dense retrievers against queries with typos?

2.1 Introduction

With the advances in neural language modeling [32], learning dense representations for text has become a vital component for many information retrieval (IR) tasks. In passage ranking and open-domain question answering, dense retrieval has become a new paradigm to retrieve relevant passages [73, 74, 94]. In contrast to traditional term-based IR models (TF-IDF and BM25) that fail to capture beyond-lexical matching, dense retrieval learns dense representations of questions and passages for semantic matching.

A typical approach for dense retrieval involves learning a dual-encoder for embedding the questions and passages [73]. A dual-encoder model consists of two separate neural networks optimized to score relevant (i.e., positive) question-passage pairs higher than irrelevant (i.e., negative) ones. At inference time, the score of a question-passage pair is computed as the inner product of the corresponding question and passage embeddings. Due to their high efficiency, dual-encoders are popular first-stage rankers in large-scale settings (in contrast to cross-encoders that can achieve higher performance, but they are not indexable and therefore are used as re-rankers [44, 143]). The whole

This chapter was published as G. Sidiropoulos and E. Kanoulas. Analysing the robustness of dual encoders for dense retrieval against misspellings. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2022.

corpus can be encoded and indexed offline, while at inference time, high-scoring passages with respect to a question can be found using efficient maximum inner product search [70].

So far, dense retrieval models have been evaluated on clean and curated datasets. However, these models will encounter user-generated noisy questions when deployed in real-life applications. Questions can include typos because of users mistyping words, such as keyboard typos (additional/missing character and character substitution), phonetic typing errors due to the close pronunciation, and misspellings. How these typos affect the encoding of questions and whether dense dual-encoder retrieval models are robust to them has not been studied yet.

Works on text classification have shown that deep neural language models such as BERT are not robust against typos [122, 150], even though they apply *WordPiece* tokenization. Ma et al. [96] and Zhuang and Zuccon [196] showed that typos can confuse even advanced BERT-based cross-encoders for re-ranking [29, 44, 115] and proposed data augmentation training for building typo-robust re-rankers. Additionally, Ma et al. [96] showed that bringing closer in the latent space the representations of the positive question-passage pairs of different questions while being far apart from negative ones can increase robustness.

While the aforementioned works studied robustness for the case of re-ranking, improving the robustness of dense retrieval for first-stage ranking has not been explored in-depth yet. Intuitively, if typos cause an inferior first-stage ranking, that will already negatively affect the performance of the re-ranker. Therefore, robustness for first-stage ranking is crucial for the overall performance. To the best of our knowledge, the work by Zhuang and Zuccon [196] is the only work that studied the first-stage ranking and used data augmentation to improve the robustness of a BERT-based Siamese encoder.

In this chapter, we study in-depth the robustness of dense retrieval for the case of dual-encoder architecture. We propose an approach that combines data augmentation with contrastive learning for robustifying dual-encoders against questions with typos. In detail, alongside augmenting questions with typos, we propose to use a contrastive loss that brings the representation of a question close to its typoed variations in the latent space while keeping it distant from other distinct questions. Here, a typoed variation of a question is a typo-augmented view of the specific question. For example, given the original question, "Where was Obama born?" a typoed variation can be "Where was Obama bprn?"

We aim to answer **RQ1**, which we break down to the following research sub-questions:

RQ1.1 Can data augmentation, contrastive learning, and their combination improve the robustness of dense retrieval to typos?

RQ1.2 Do certain typoed words affect the robustness of the question encoding more than others?

RQ1.3 Do the proposed method improve the robustness of the question encoding by ways other than simply learning to ignore the typoed word?

Our main contributions are the following: (i) we propose an approach for robustifying dense retrievers towards typos in user questions that combines data augmentation

Table 2.1: Number of questions in each dataset, and the average length of question.

	Train	Dev	Test	Avg. question length
MS MARCO	502,939	6,980	6,837	5.94
Natural Questions	79,168	8,757	3,610	9.20

with contrastive learning and performs better than applying each component separately, (ii) we perform a thorough analysis of the robustness of dense retrieval, and (iii) show that typos in various words influence performance differently.[1]

2.2 Experimental Setup

In this section, we discuss the datasets, the metrics, and the robustness methods we experiment with to answer our research questions.

2.2.1 Datasets

We conduct our experiments on two large-scale datasets, namely, MS MARCO passage ranking [113] and Natural Questions [78]. In MS MARCO passage ranking, the goal is to rank passages based on their relevance to a question (i.e., the probability of including the answer). The data collection consists of 8.8 million passages; the questions were selected from Bing search logs. Natural Questions is a large-scale dataset for open-domain QA over Wikipedia, and its questions were selected from Google search logs. Table 2.1 shows the statistics of the two datasets.

2.2.2 Metrics

To measure the retrieval performance on MS MARCO, we use the official metric MRR (@10) alongside the commonly reported Recall (R) at top-k ranks [74, 124].[2] Following previous work on Natural Questions, we use answer recall (AR) at the top-k retrieved passages [73, 124]. Answer recall measures whether at least one of the top-k retrieved passages contains the ground-truth answer.

2.2.3 Methods

In this section, we present the three approaches we apply as extensions to the baseline model in order to increase robustness. Below we describe the dual-encoder model we use for our experiments:

- **Dense Retriever (DR)** is a dual-encoder BERT-based model used for scoring question-passage pairs [73]. Given a question q, a relevant passage p^+ and a set of irrelevant

[1] `https://github.com/GSidiropoulos/dense-retrieval-against-misspellings`

[2] Similar to previous work, we report the metrics on MSMARCO (Dev) since the correct answers for the test set are not available to the public.

passages $\{p_1^-, p_2^-, \ldots, p_n^-\}$, the model learns to rank p^+ higher than the negative passages via the optimization of the negative log-likelihood of the relevant passage:

$$\begin{aligned} & \mathcal{L}_1(q_i, p_i^+, p_{i,1}^-, \ldots, p_{i,n}^-) \\ = \quad & -\log \frac{e^{\text{sim}(q_i, p_i^+)}}{e^{\text{sim}(q_i, p_i^+)} + \sum_{j=1}^{n} e^{\text{sim}(q_i, p_{i,j}^-)}}. \end{aligned} \tag{2.1}$$

- **Data Augmentation (DR + Data augm.)** is one of the traditional approaches for robustifying neural models. By exposing DR to questions with and without typos, the model learns to be invariant to typos. Similar to Zhuang and Zuccon [196], for each original correctly written question, on training time, we draw an unbiased coin. If the result is heads, we use the original question for training. If the result is tails, we use one of its typoed variations.

- **Contrastive learning (DR + CL)** of representations works by maximizing the agreement between differently augmented views of the same object. We propose a contrastive loss that compares the similarity between a question and its typoed variations and other distinct questions. In contrast with data augmentation, which explicitly trains on typoed question-passage pairs, here such pairs are seen implicitly only. In detail, in addition to Equation 2.1, we introduce a loss that enforces that a question q and its typoed variations q^+ are close together in the latent space, while being far apart from other distinct questions $\{q_1^-, q_2^-, \ldots, q_n^-\}$:

$$\begin{aligned} & \mathcal{L}_2(q_i, q_i^+, q_{i,1}^-, \cdots, q_{i,n}^-) \\ = \quad & -\log \frac{e^{\text{sim}(q_i, q_i^+)}}{e^{\text{sim}(q_i, q_i^+)} + \sum_{j=1}^{n} e^{\text{sim}(q_i, q_{i,j}^-)}}. \end{aligned} \tag{2.2}$$

  The final loss is a weighted average of the two losses:

$$\mathcal{L} = w_1 \cdot \mathcal{L}_1 + w_2 \cdot \mathcal{L}_2. \tag{2.3}$$

  The weights w_1 and w_2 are hyper-parameters and therefore need to be defined. Giving equal weights to the two losses is an effective and straightforward combination method that we used in our experiments.

- **Combination (DR + Data augm. + CL)** method consists of data augmentation combined with contrastive learning. Specifically, alongside augmenting questions with typos we propose to use the contrastive loss of Equation 2.2 that brings the representation of a question close to its typoed variations while keeping it distant from other distinct questions. The final loss is a weighted average of the three losses:

$$\mathcal{L} = w_1 \cdot \mathcal{L}_1 + w_2 \cdot \mathcal{L}_2 + w_3 \cdot \mathcal{L}_3, \tag{2.4}$$

  where $\mathcal{L}_3$ represents the data augmentation and is computed similarly to Equation 2.1 but for the typoed variation q^+ of the original question q. For our experiments, we use an equal weighting setting for the weights w_1,w_2, and w_3.

2.2.4 Simulating typos

To assess the robustness of the proposed methods, a large-scale dataset for passage retrieval with typoed questions is necessary. Unfortunately, such a dataset does not exist, and therefore we build one by simulating typos over the original Natural Questions and MS MARCO datasets. In detail, we simulate typos produced by humans by augmenting the original questions in the dataset with synthetically generated typoed ones.

In order to simulate typos, we apply the following transformations that often occur in human-generated questions [see, e.g., 9, 52, 196].

- Random: Inserts, deletes, swaps, or substitutes a random character, e.g., *committee* $\rightarrow$ {*copmmittee*, *commttee*, *comimttee*, *commitlee*}.
- Keyboard: Swaps a random character with those close to each other on the QWERTY keyboard, e.g., *committee* $\rightarrow$ *comnittee*.
- Common misspellings: Replaces words with misspelled ones, defined in a dictionary of common user-generated misspellings, e.g., *committee* $\rightarrow$ *comittee*.

2.2.5 Implementation details

The DR model used in our experiments is trained using the in-batch negative setting described in [73]. The question and passage BERT encoders are trained for $50K$ steps, with a batch size of 48. The learning rate is set to $2e$-5 using Adam, and the rate of the linear scheduling with a warm-up is set to 0.1. Moreover, we use the same hyper-parameters for the three robustifying methods described in Section 2.2.3, in order to ensure a fair comparison.

For generating typos in the training phase as well as building the typo-robustness test set, we use the open-source Aug library [9]. Each word in a question gets transformed with a probability of 0.2, and the transformation type (Section 2.2.4) gets chosen at random.

2.3 Results

In this section, we present our experimental results that answer our research questions. We aim to answer **RQ1.1** by comparing the retrieval performance of the methods we consider (Section 2.2.3) for two settings: clean questions and questions with typos. In Table 2.2, we observe that on clean questions, data augmentation as well as our two proposed approaches, namely, contrastive learning and data augmentation combined with contrastive learning, do not harm the performance. Moreover, all the approaches for robustifying DR are performing significantly better than the original DR, on questions with typos. That indicates that they successfully robustify the underlying dual-encoder model. That said, our proposed data augmentation combined with contrastive learning approach holds the best performance.

Following previous work, we randomly introduce typos to questions. However, we want to investigate if the performance of the approaches we consider remains the same irrespectively of the word in which the typo appears. To answer **RQ1.2**, we create two

Table 2.2: Retrieval results for the settings of (i) clean questions (Original), and (ii) questions with typos (Typos in Random Words). Statistical significance difference with paired t-test ($p < 0.05$) DR=d; DR+ Data augm.=a; DR+CL=c.

	Natural Questions (Test)					
	Original			Typos in Random Words		
	AR@5	AR@20	AR@100	AR@5	AR@20	AR@100
DR	67.31	78.22	85.42	49.52	63.98	76.12
DR + Data augm.	66.45	79.03	85.56	60.69	73.76	**83.37**
DR + CL (ours)	66.31	77.42	85.45	55.51	69.27	80.52
DR + Data augm. + CL (ours)	67.47	78.83	85.67	**62.13**^{dac}	**74.87**^{dac}	83.26^{dc}
	MS MARCO (Dev)					
	Original			Typos in Random Words		
	MRR@10	R@50	R@1000	MRR@10	R@50	R@1000
DR	28.11	73.46	93.36	15.11	46.47	74.02
DR + Data augm.	28.26	72.66	93.07	22.00	61.68	86.49
DR + CL (ours)	28.95	73.01	93.64	19.37	55.08	80.69
DR + Data augm. + CL (ours)	29.14	73.85	93.69	**22.84**^{dac}	**63.21**^{dac}	**87.52**^{dac}

additional test settings for the case of questions with typos. Specifically, we create (i) a setting where typos appear only in non-stopwords, and (ii) a setting where typos appear only in utterances with a lexical match with the relevant passage.[3] In detail, we consider the overlapping consecutive words between the ground-truth passage and the question (e.g., "Who was the president of the united states during wwi?", and "Woodrow Wilson, a leader of the Progressive Movement, was the 28th President of the United States (1913-1921). After a policy of neutrality at the outbreak of World War I, he led America into war." mark the "president of the united states" as available utterance to introduce typos). The highly discriminative utterances obtained through this heuristic are typically entity mentions.

As we can see in Table 2.3 and by comparing the numbers with the results in Table 2.2, the effectiveness of the methods varies across the three settings. Particularly, robustness deteriorates when typos do not appear randomly. In detail, the most significant losses occur when typos appear on discriminative utterances. Our proposed data augmentation combined with contrastive learning approach remains the best-performing one across all settings.

To better understand the discrepancy in robustness between the three settings of questions with typos, we conduct the following analysis. For the setting where typos randomly appear on questions, we study how the frequency on the training set of the typoed words at test time affects robustness. As shown in Figure 2.1, there is a strong

[3]We build the new settings using the same probability for introducing typos (Section 2.2.5), and we do not retrain the models on the new settings.

Table 2.3: Retrieval results for the settings of (i) questions with typos in non-stopwords (Typos in Non-stopwords), and (ii) questions with typos in highly discriminative utterances (Typos in Discriminative Utterances). Statistical significance difference w/ paired t-test ($p < 0.05$) DR=d; DR+Data augm.=a; DR+CL=c.

	Natural Questions (Test)					
	Typos in Non-stopwords			Typos in Discriminative Utterances		
	AR@5	AR@20	AR@100	AR@5	AR@20	AR@100
DR	40.60	55.48	68.72	38.89	53.37	68.00
DR + Data augm.	56.14	69.94	80.19	51.68	66.12	78.08
DR + CL (ours)	49.47	64.12	76.48	44.04	59.00	72.43
DR + Data augm. + CL (ours)	**57.78**^{dac}	**70.77**^{dac}	**81.38**^{dac}	**53.15**^{dac}	**67.28**^{dac}	**78.61**^{dac}
	MS MARCO (Dev)					
	Typos in Non-stopwords			Typos in Discriminative Utterances		
	MRR@10	R@50	R@1000	MRR@10	R@50	R@1000
DR	11.83	38.98	66.16	10.51	34.17	59.71
DR + Data augm.	18.51	54.82	81.92	16.51	49.06	77.47
DR + CL (ours)	15.44	46.69	73.27	12.44	39.17	66.69
DR + Data augm. + CL (ours)	**19.47**^{dac}	**56.22**^{dac}	**83.61**^{dac}	**17.58**^{dac}	**50.81**^{dac}	**79.51**^{dac}

connection between the frequency of the typoed words and the retrieval performance. As the frequency of the typoed words decreases, the performance drops significantly. To this extent, our proposed data augmentation combined with contrastive learning approach remains the best performing one, with the performance gap increasing as the frequency of the typoed word decreases. The results in Figure 2.1 can also explain why we observe the highest losses in performance on the setting with typos in discriminative utterances. In general, the discriminative utterances (entity mentions) diversity between the dataset splits is higher compared to other words appearing in questions (e.g., interrogative, linking words). For instance, given the question "When was Barack Obama born?" at the train set, during test time, we are more likely to encounter a similar question where the main entity changes, e.g., "When was Barack Obama born?" rather than a paraphrase of the question keeping the same entity, e.g., "What is Obama's birthday?".

For **RQ1.3**, we consider a simple baseline where the typoed words are identified and removed from the question before being fed to the original DR model. We compare our best-performing approach against the aforementioned baseline. If our proposed approach only learns to ignore words with typos, then we argue that the performance of the baseline should be competitive to ours. In many cases, removing the typoed word can be a valid approach since the importance of words in a question varies. For instance, considering the question "Where was president Lincoln born?" we see that "Lincoln" is crucial for the meaning of the question while "was" adds no information. With that in mind, we study **RQ1.3** with respect to the relative importance of the typoed words

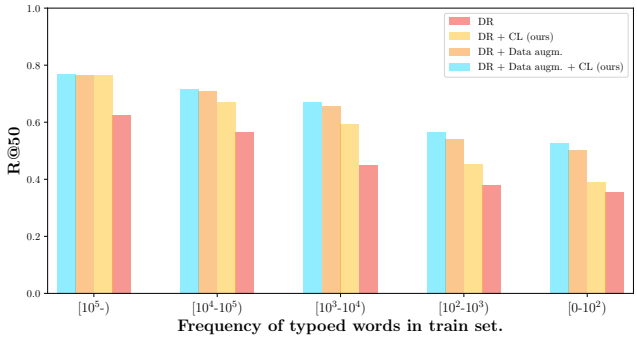


Figure 2.1: Retrieval performance (Average Recall@50) w.r.t. the frequency on the training set, of the typoed words at test-time; on MS MARCO (Dev). Questions are split into bins w.r.t. the frequency of their typoed words.

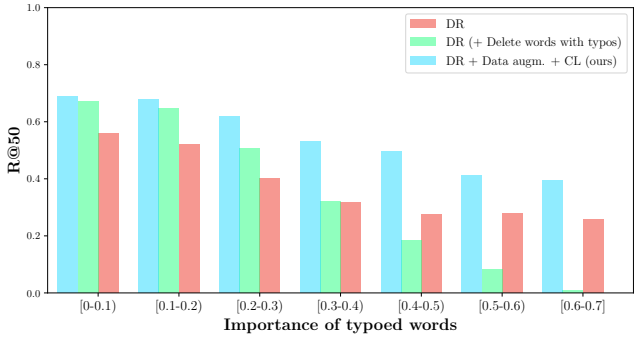


Figure 2.2: Retrieval results w.r.t. the relevant importance of the typoed words; on MS MARCO (Dev). Questions are split into bins w.r.t. the relevant importance of their typoed words.

within the question.[4]

Our method does not simply learn to ignore words with typos since, as we can see from Figure 2.2, it consistently outperforms the baseline. Furthermore, Figure 2.2 highlights that when the importance of typoed words (as measured using R@50) is low, simply ignoring them is a highly competitive approach. On the other hand, as the importance of the typoed words increases, the effectiveness of just ignoring these words decreases dramatically, to the extent that keeping the typoed words performs better. That can be attributed to the application of the WordPiece tokenizer (by the underlying BERT model) that allows DR to recover from some typos, such as when the character n-gram splits remain intact despite the typos. For instance original *robustness* and typoed *robustnessd* will be split into *[robust, ##ness]* and *[robust, ##ness, ##d]* respectively.

[4]We define a word's relevant importance as the ratio of its IDF to the sum of the IDFs of every word in the question.

2.4 Conclusions

In this chapter, we provided insights into the robustness of dual-encoders for dense retrieval when dealing with typos in user questions. We proposed an approach for robustifying dual-encoders that combines data augmentation with contrastive learning. Our experimental results showed that our proposed method not only improves robustness but also performs better than separately applying data augmentation or contrastive learning. Analysis of the methods we explored showed that typos in various words do not influence performance equally. In particular, typos on words that are less frequent on the training set and more important for a question are harder to address. Our proposed technique remains the best-performing one in these settings, however, the performance deteriorates significantly compared to a clean question.

There is a significant question that has arisen throughout our study: What could a dual-encoder actually learn to fix the problem? The *WordPiece* tokenizer, applied by BERT, allows models to recover from some typos. However, it would be ideal if embeddings could be learned at the character n-gram level to allow recovery from typical character substitution, deletion, etc. Furthermore, word-to-word interactions during training (e.g., through a late interaction model [74]) could also allow implicitly to learn the "correct spelling" of a typoed word during training. We leave these directions as future work.

In this chapter, we proposed to robustify dense retrievers via data augmentation and an additional robustifying subtask that aims to align the representations of the typo-free question and its typoed variant. While we have multiple different typo-augmented views of the same question, since we rely on synthetic noise generation, we only used a single typoed variant per question at each step. In the next chapter, we argue that multiple typoed variants can be used simultaneously to boost robustness further.

3

Contrastive Learning with Multiple Positives for Typo-robust Dense Retrieval

In the previous chapter–similar to other work on improving the robustness of dense retrievers against typos–we combined (i) data augmentation to obtain typoed queries during training time with (ii) additional robustifying subtasks that aim to align the original, typo-free queries with their typoed variants. Even though multiple typoed variants are available as positive samples per query, the aforementioned family of methods assumes a single positive sample and a set of negative ones per anchor and tackles the robustifying subtask with contrastive learning. Therefore, the approach from the previous chapter makes insufficient use of the multiple positives (typoed queries). In this chapter, we aim to answer **RQ2**: Can we improve the robustness of dense retrievers with contrastive learning in a way that accounts for multiple positives and negatives?

3.1 Introduction

Dense retrieval has become the new paradigm in passage retrieval. It has demonstrated higher effectiveness than traditional lexical-based methods due to its ability to tackle the vocabulary mismatch problem [73]. Even though dense retrievers are highly effective on typo-free queries, they can witness a dramatic performance decrease when dealing with queries that contain typos [139, 144, 197]. Recent works on robustifying dense retrievers against typos use data augmentation to obtain typoed versions of the original queries at training time. Moreover, they introduce additional robustifying subtasks to minimize the representation discrepancy between the original query and its typoed variants.

Sidiropoulos and Kanoulas [139] applied an additional contrastive loss to enforce the latent representations of the original, typo-free queries to be closer to their typoed variants. Zhuang and Zuccon [197] used a self-teaching training strategy to minimize the difference between the score distribution of the original query and its typoed variants. Alternatively, Tasawong et al. [153] employed dual learning in combination with self-teaching [197] and contrastively trained the dense retriever on the prime task of passage

This chapter was published as G. Sidiropoulos and E. Kanoulas. Improving the robustness of dense retrievers against typos via multi-positive contrastive learning. In *Advances in Information Retrieval - 46th European Conference on Information Retrieval, ECIR, Proceedings, Part III*. Springer, 2024.

retrieval and the dual task of query retrieval (learns the query likelihood to retrieve queries for passages).

Despite the improvements in robustness, existing typo-robust methods do not always make optimal use of the available typoed queries. They address the robustifying subtasks with contrastive learning, assuming a single positive sample (query) and a set of negative ones per anchor (depending on the approach, the anchor can be either a query or a passage). However, alongside the original query, its multiple typoed variants are available. Hence, there is more than one positive sample per anchor. As a result, we can use all the available positives simultaneously and apply multi-positive contrastive learning instead (i.e., contrastive learning that supports multiple positives). For instance, Tasawong et al. [153] compute the contrastive loss for the query retrieval subtask using only the original, typo-free query as relevant for a given passage. Given a passage, we argue that both the original query and its typoed variations can be considered as relevant and adopt a multi-positive contrastive loss instead.

Literature on contrastive learning has shown that including multiple positives can enhance the ability of the model to discriminate between signal and noise (negatives) [75, 101]. Intuitively, multiple negatives focus on what makes the anchor and the negatives dissimilar, while multiple positives focus on what makes the anchor and the positives similar. To this end, contrasting among multiple positives and negatives can bring an anchor and all its positives closer together in the latent space while keeping them far from the negatives.

In this chapter, we revisit recent methods in typo-robust dense retrieval and unveil that, in many cases, they do not sufficiently use the multiple positives that are available. Specifically, when tackling the robustifying subtasks, they ignore that multiple positives are available per anchor and consider contrastive learning with a single positive. In contrast, we suggest using all the available positives and adopting a multi-positive contrastive learning approach. We aim to answer the following research sub-questions in order to tackle **RQ2**:

RQ2.1 Can our multi-positive contrastive learning approach increase the robustness of dense retrievers that use contrastive learning with a single positive?

RQ2.2 Does our multi-positive contrastive learning variant outperform its single-positive counterpart regardless of the number of positives?

Our experimental results on two datasets show that our proposed approach of employing multi-positive contrastive learning yields improvements in robustness compared to contrastive learning with a single positive.[1]

3.2 Methodology

Contrastive learning is a vital component for training an effective dense retriever. Current typo-robust dense retrievers use contrastive learning with a single positive sample and multiple negative ones for both the main task of passage retrieval and the robustifying subtasks. In detail, given an anchor x, a positive sample x^+, and a set of

[1] https://github.com/GSidiropoulos/typo-robust-multi-positive-DR

negative samples X^-, the contrastive prediction task aims to bring the positive sample closer to the anchor than any other negative sample:

$$\mathcal{L}_{CE}(x, x^+, X^-) = -\log \frac{e^{f(x,x^+)}}{e^{f(x,x^+)} + \sum\limits_{x^- \in X^-} e^{f(x,x^-)}}, \tag{3.1}$$

where f is a similarity function (e.g., dot product).

However, in many cases, multiple positive samples are available per anchor and can be used simultaneously to increase the discriminative performance of the model. As opposed to the aforementioned contrastive loss that supports a single positive, we propose employing a multi-positive contrastive loss to benefit from all the available positives. Given an anchor x, multiple positive samples X^+, and multiple negatives X^-, a multi-positive contrastive loss [75] is computed as:

$$\mathcal{L}_{MCE}(x, X^+, X^-) = -\frac{1}{|X^+|} \sum_{x^+ \in X^+} \log \frac{e^{f(x,x^+)}}{e^{f(x,x^+)} + \sum\limits_{x^- \in X^-} e^{f(x,x^-)}}. \tag{3.2}$$

This chapter aims to identify cases in typo-robust dense retrieval methods where the robustifying subtasks consider only a single positive sample, even though multiple ones are available, and optimize a contrastive loss. Next, we replace the contrastive loss with its multi-positive alternative to benefit from all the available positives. Below we present the typo-robust dense retrieval methods we build upon followed by our multi-positive variants. We focus on dense retrievers that follow the dual-encoder architecture [73]. A traditional dense retriever, **DR**, is optimized only with the passage retrieval task. Given a query q, a positive/relevant passage p^+, and a set of negative/irrelevant passages $P^- = \{p_i^-\}_{i=1}^N$, the learning task trains the query and passage encoders via minimizing the softmax cross-entropy: $\mathcal{L}_{CE}^p = \mathcal{L}_{CE}(q, p^+, P^-)$. Positive query-passage pairs are encouraged to have higher similarity scores and negative pairs to have lower scores.

3.2.1 Dense retriever with self-supervised contrastive learning

DR+CL alternates DR with an additional contrastive loss that maximizes the agreement between differently augmented views of the same query [139]. This loss enforces that a query q and its typoed variation q', sampled from a set of available typoed variations $Q' = \{q_i'\}_{i=1}^K$, are close together in the latent space and distant from other distinct queries $Q^- = \{q_i^-\}_{i=1}^M$: $\mathcal{L}_{CE}^t = \mathcal{L}_{CE}(q, q', Q^-)$. The final loss is computed as a weighted summation, $\mathcal{L} = w_1\mathcal{L}_{CE}^p + w_2\mathcal{L}_{CE}^t$.

DR+CL$_M$ is our multi-positive variant of DR+CL. Given a query q, instead of sampling a different typoed variant q' from a set Q' at each update, we propose simultaneously employing all typoed variants. To do so, we replace $\mathcal{L}_{CE}^t$ with the following multi-positive contrastive loss that accounts for multiple positives: $\mathcal{L}_{MCE}^t = \mathcal{L}_{MCE}(q, Q', Q^-)$. The final loss is: $\mathcal{L} = w_1\mathcal{L}_{CE}^p + w_2\mathcal{L}_{MCE}^t$.

3.2.2 Dense retriever with dual learning

DR+DL trains a robust, dense retriever via a contrastive dual learning mechanism [90]. In contrast to classic DR, which is optimized for passage retrieval only ($\mathcal{L}_{CE}^p$), DR+DL

is optimized for the prime task of passage retrieval (i.e., learns to retrieve relevant passages for queries) and the dual task of query retrieval (i.e., learns to retrieve relevant queries for passages). Therefore, given a passage p, a positive query q^+, and a set of negative queries $Q^- = \{q_i^-\}_{i=1}^M$, it further minimizes the loss for the dual task: $\mathcal{L}_{CE}^q = \mathcal{L}_{CE}(p, q^+, Q^-)$. The dual training loss is added to the prime training loss to conduct contrastive dual learning and train the dense retriever. Specifically, the final loss is computed as $\mathcal{L} = \mathcal{L}_{CE}^p + w\mathcal{L}_{CE}^q$, where w is used to weight the dual task loss.
DR+DL$_M$ is our multi-positive variant of DR+DL. Contrary to DR+DL, we propose that for the query retrieval task, given a passage p, we can have a set of relevant queries consisting of the typo-free query and its typoed variants, $\mathcal{Q} = \{q^+, q'_1, q'_2, \ldots, q'_K\}$. Thus, we replace the contrastive loss of $\mathcal{L}_{CE}^q$ with a multi-positive contrastive loss, which can account for multiple relevant queries at the same time. We define the multi-positive contrastive loss for the dual task as: $\mathcal{L}_{MCE}^q = \mathcal{L}_{MCE}(p, Q, Q^-)$. The final loss is computed as $\mathcal{L} = \mathcal{L}_{CE}^p + w\mathcal{L}_{MCE}^q$.

3.2.3 Dense retriever with dual learning and self-teaching

DR+ST+DL trains a dense retriever with dual learning and self-teaching [153]. Similar to DR+DL, it minimizes the $\mathcal{L}_{CE}^p$ and $\mathcal{L}_{CE}^q$ losses for the main task of passage retrieval and the subtask of query retrieval, respectively. The additional self-teaching mechanism distills knowledge from a typo-free query q into its typoed variants $Q' = \{q'_i\}_{i=1}^K$ by forcing the model to match score distributions of misspelled queries to the score distribution of the typo-free query for both the passage retrieval and query retrieval task. This is achieved by minimizing the KL-divergence losses: (i) $\mathcal{L}_{KL}^p = \frac{1}{K}\sum_{k=1}^K \mathcal{L}_{KL}(s_p'^k, s_p)$, where $\{s_p'^1, s_p'^2, \ldots, s_p'^K\}$ and s_p is the score distribution in a passage-to-queries direction (passage retrieval) for the typoed queries and the typo-free query, respectively, and (ii) $\mathcal{L}_{KL}^q = \frac{1}{K}\sum_{k=1}^K \mathcal{L}_{KL}(s_q'^k, s_q)$, where $\{s_q'^1, s_q'^2, \ldots, s_q'^K\}$ and s_q is the score distribution in a query-to-passages direction (query retrieval) for the typoed queries and the typo-free query, respectively. The final loss is computed as the weighted summation of the four losses, $\mathcal{L} = (1-\beta)((1-\gamma)\mathcal{L}_{CE}^p + \gamma\mathcal{L}_{CE}^q) + \beta((1-\sigma)\mathcal{L}_{KL}^p + \sigma\mathcal{L}_{KL}^q)$.
DR+ST+DL$_M$ is our multi-positive variant of DR+ST+DL. Even though DR+ST+DL simultaneously uses all the available typo variations of a query in order to calculate the KL divergence losses for the prime passage retrieval task and the dual query retrieval, it uses only the typo-free query to compute the contrastive loss for query retrieval ($\mathcal{L}_{CE}^q$). To fully benefit from the multiple available typoed queries per typo-free query, we replace the contrastive loss for query retrieval $\mathcal{L}_{CE}^q$ with a multi-positive variant that supports samples with multiple positives $\mathcal{L}_{MCE}^q$. The final loss is computed as the weighted summation, $\mathcal{L} = (1-\beta)((1-\gamma)\mathcal{L}_{CE}^p + \gamma\mathcal{L}_{MCE}^q) + \beta((1-\sigma)\mathcal{L}_{KL}^p + \sigma\mathcal{L}_{KL}^q)$.

3.3 Experimental Setup

Query augmentation From the aforementioned methods, those employing queries with typos in their training scheme are augmentation-based. During training, the typoed queries are generated from the original, typo-free queries through a realistic typo generator [112]. The typo generator applies the following transformations that often

occur in human-generated queries: random character insertion, deletion and substitution, swapping neighboring characters, and keyboard-based character swapping [52].

Datasets and evaluation We conduct our experiments on MS MARCO passage ranking [113] and DL-Typo [197] on their typo-free and typoed versions. Both datasets use the same underlying corpus of 8.8 million passages and $\sim 400K$ training queries but differ in evaluation queries. DL-Typo provides 60 real-world queries with typos alongside their manually corrected typo-free version. The development set of MS MARCO consists of $6,980$ queries (the test set is not publicly available). Following previous works [153, 197], we obtain typo variations for each typo-free query via a synthetic typo generation model and repeat the typo generation process 10 times. To measure the retrieval performance, we report the official metrics on each dataset. For the evaluation on the typo version of MS MARCO, we report the metrics averaged for each repeated experiment since typoed queries are generated 10 times for each original query.

Implementation details We follow an in-batch negative training setting with 7 hard negative passages per query and a batch size of 16 to train the dense retrievers.[2] We use AdamW optimizer with a 10^{-5} learning rate, linear scheduling with $10K$ warm-up steps, and decay over the rest of the training steps. We train up to $150K$ steps. We implement the query and passage encoders with BERT [32]. When applicable, we set the query augmentation size to 40. For the remaining hyperparameters specific to each method (e.g., weight w in DR+CL), we use the values initially proposed by the creators of each method. We use the Tevatron toolkit [47] to train the models and the Ranx library [6] to evaluate the retrieval performance. Finally, we use the typo generators from the TextAttack toolkit [112] for all the methods we experiment with to augment the training queries.

3.4 Results

To answer **RQ2.1**, we compare the retrieval performance of our multi-positive contrastive learning approaches against the original models. From Table 3.1, we see that employing our multi-positive contrastive learning approach yields improvements in robustness against typos upon the original methods that use contrastive learning with a single positive.

As expected, the more dramatic improvement comes when applying multi-positive contrastive learning on DR+DL since the original work only considers the typo-free query as positive when computing the contrastive loss for query retrieval (see Section 3.2.2). In contrast, in our DR+DL$_M$, we consider the typo-free query and all its available typoed variants as positives and use a multi-positive contrastive loss for query retrieval. We also see improvements when comparing DR+CL vs. our DR+CL$_M$. In detail, employing all available positives (typoed queries) at once and using multi-positive

[2]The original methods and our proposed counterparts employ the same number of original, typo-free query-passage pairs per batch. However, our method leverages multiple typoed variants for each query; therefore, the batch we need to fit in the GPU memory is larger.

Table 3.1: Retrieval results for the settings of (i) clean queries (Clean), and (ii) queries with typos (Typo). Multi-positive CL indicates the use of multi-positive contrastive loss. Statistical significant gains (two-tailed paired t-test with Bonferroni correction, $p < 0.05$) obtained from models with multi-positive contrastive loss (ours) over their original version with standard contrastive loss are indicated by †.

Model	Multi-positive CL	MS MARCO				DL-Typo					
		Clean		Typo		Clean			Typo		
		MRR@10	R@1000	MRR@10	R@1000	nDCG@10	MRR	MAP	nDCG@10	MRR	MAP
DR	✗	.331	.953	.140	.698	.677	.850	.555	.264	.395	.180
DR+DL	✗	.332	.953	.140	.698	.679	.826	.557	.269	.411	.186
DR+DL$_M$	✓	.335	.958	.213†	.866†	.699	.864	.585	.347†	.452	.259†
DR+CL	✗	.321	.957	.170	.787	.659	.797	.535	.284	.411	.207
DR+CL$_M$	✓	.322	.956	.178†	.811†	.652	.847	.539	.290	.447	.215
DR+ST+DL	✗	.334	.951	.259	.893	.681	.868	.567	.412	.543	.315
DR+ST+DL$_M$	✓	.335	.955	.261	.902†	.687	.870	.579	.426†	.583	.342†

contrastive loss outperforms sampling a different positive at each update and using a single positive contrastive loss (see Section 3.2.1). The improvements are held even when comparing our DR+DL+ST$_M$ against DR+DL+ST, a model that already uses multiple positives. As seen in Section 3.2.3, DR+DL+ST uses a contrastive loss with a single positive for the query retrieval dual task (i.e., $\mathcal{L}^q_{CE}$) while considering multiple positives simultaneously to compute the KL-divergence losses (i.e., $\mathcal{L}^p_{KL}$, $\mathcal{L}^q_{KL}$).

At this point, we want to explore how the different numbers of positives affect our multi-positive approach (**RQ2.2**). To do so, we compare our DR+DL+ST$_M$ against DR+DL+ST. In its training, the latter already employs multiple positives simultaneously to compute the KL-divergence losses. However, our multi-positive approach fully benefits from the multiple available positives by incorporating them when computing the contrastive loss for query retrieval ($\mathcal{L}^q_{CE} \rightarrow \mathcal{L}^q_{MCE}$). Table 3.2 unveils that our multi-positive variant consistently outperforms the original model for the different numbers of typoed variants per query. In addition, our findings suggest that increasing K results in improved performance, but only up to a certain threshold, beyond which the performance stabilizes. Specifically, we observe no additional performance improvement beyond 30 augmentations. Furthermore, the most significant performance difference occurs when augmentations are increased from 1 to 10; subsequent increases yield only marginal gains.

Table 3.2: Retrieval results for different query augmentation sizes (K). We report the results in the format "$R@1000$ ($MRR@10$)" on MS MARCO with typos.

	Multi-positive contrastive loss	K				
		1	10	20	30	40
DR+ST+DL	✗	.884 (.251)	.892 (.258)	.894 (.258)	.893 (.259)	.893 (.259)
DR+ST+DL$_M$	✓	**.884 (.251)**	**.898 (.260)**	**.900 (.260)**	**.902 (.261)**	**.902 (.261)**

3.5 Conclusions

In this chapter, we revisit recent studies in typo-robust dense retrieval and showcase that they do not always make sufficient use of multiple positive samples. They assume a single positive sample and multiple negatives per anchor and use contrastive learning for the robustifying subtasks. In contrast, we propose to leverage all the available positives and employ multi-positive contrastive learning. Experimentation on two datasets shows that following a multi-positive contrastive learning approach yields improvements in the robustness of the underlying dense retriever upon contrastive learning with a single positive.

At this point, we list some of the limitations of our work that can serve as directions for future research. As we saw in Table 4, increasing the number of augmentations can improve performance. However, there is a trade-off between performance and computation resources. The multiviewed batch (i.e., the augmented batch that contains all the available typo-augmented views of the queries in the batch) used by methods that use multiple positives is larger compared to the batch of the respective base method that uses only a single positive. Future research could look into creating more efficient ways to train while using multiple negatives simultaneously. Moreover, in this chapter, we used BERT as the backbone of all the dense retrievers we experimented with. However, recent work has shown that using CharacterBERT as the backbone improves robustness against typos [197]. Future work can explore the combination of a dense retriever that uses CharacterBERT and is trained with multi-positive contrastive learning.

In this chapter, similar to the previous Chapter 2, we set out to robustify dense retrievers by learning better representations for the typoed text. In the next chapter, we move on to study if uncertainty-aware dense retrievers are robust against typos.

4

Modeling Uncertainty in Dense Retrieval

Chapters 2 and 3 focused on improving robustness against typos via learning better representations for noisy text. However, this is not the only method for building robust models. Alternatively, uncertainty estimation can also be considered. Uncertainty-aware retrieval systems tend to be more robust to data distribution shifts.

The current paradigm in dense retrieval is to represent queries and passages as low-dimensional real-valued vectors using neural language models, and then compute query-passage similarity as the dot product of these vector representations. A limitation of this approach is that these learned representations cannot capture or express uncertainty. The multivariate representation learning (MRL) framework proposed by Zamani and Bendersky [187] is the first method that works in the direction of modeling uncertainty in dense retrieval. This framework represents queries and passages as multivariate normal distributions, and computes query-passage similarity as the negative Kullback-Leibler (KL) divergence between these distributions. Furthermore, MRL formulates KL divergence as a dot product, allowing for efficient first-stage retrieval using standard maximum inner product search. In this chapter, we aim to answer **RQ3**: Can MRL capture uncertainty in queries that contain typos?

Owing to the source code and pre-trained models being unavailable in the work by Zamani and Bendersky [187], answering RQ3 involves a reproducibility study of the original work. We attempt to reproduce MRL on an academic computational budget (i.e., under memory constraints). In particular, we aim to uncover the extent to which the reported results and trends of the original work hold under a memory-limited, single GPU setup.

4.1 Introduction

Dense retrieval has become the new paradigm in first-stage retrieval, largely replacing lexical methods which cannot model semantic information as well as neural models. Dense retrievers following the dual-encoder architecture [73] are popular first-stage retrievers due to their performance and scalability. This paradigm uses pre-trained neural language models to encode queries and passages as low-dimensional real-valued

This chapter was published as G. Sidiropoulos, S. Bhargav, P. Eustratiadis, and E. Kanoulas. Multivariate dense retrieval: A reproducibility study under a memory-limited setup. *Under submission*.

dense vectors, with relevance defined as their dot product. Passages are encoded offline and stored in a dense index. At query time, retrieval can be done efficiently using maximum inner product search (MIPS). However, representing queries and passages as single vectors has an important limitation that has influenced the research landscape: these representations do not model, capture, or express predictive uncertainty [160]. At the same time, there are various sources of uncertainty arising both from the data and the neural retrieval models:

- **Query uncertainty.** User queries may include misspellings, ambiguity, and incomplete or inaccurate information (e.g., false memories). Furthermore, in a realistic setting, the retrieval system has minimal to no prior knowledge about the distribution of the queries, except for possibly a few assumptions (e.g., the language they are in, or common user mistakes).

- **Passage uncertainty.** Passages may present similar uncertainty-inducing artifacts to queries, such as misspellings and ambiguity. Unlike queries, the retrieval model has little prior knowledge of the passage collection as a whole, i.e., only of the documents used during training.

- **Relevance uncertainty.** Relevance, or ranking uncertainty refers to the confidence of the model in the estimated query-passage relevance. Such an estimator may be anything between a deterministic function of query and passage uncertainty, e.g., the model reproduced in this chapter, to a stochastic function of deterministic query and passage representations, e.g., a Monte-Carlo dropout Bayesian estimator [25].

Uncertainty estimation remains largely unexplored for the case of first-stage dense retrieval, despite it having received increased attention from the community in the case of re-ranking [25, 39, 61, 166, 195]. Recently, Zamani and Bendersky [187] proposed the multivariate representation learning (MRL) framework, the first approach that models uncertainty in the context of dense retrieval. MRL uses predictive variance as a proxy for uncertainty. Each query and passage is mapped to a multivariate Gaussian distribution parameterized by a mean vector and a (diagonal) covariance matrix, where the mean represents the predicted query or passage embedding, and the variance represents the uncertainty of said embedding. However, different from existing approaches to modelling uncertainty in IR that use Bayesian inference [25], variance in the MRL framework does not express statistical variance, i.e., deviation from the mean, as much as it expresses predicted risk. In essence, this is a trade-off between being theoretically principled and computationally efficient. Admittedly, computational efficiency is of the utmost importance in first-stage retrieval, where one has to manage collections of potentially billions of passages.

Having represented queries and passages as multivariate normal distributions, the authors of the original paper proceed to formulate a query-passage relevance scoring function based on a simplified version of the Kullback–Leibler (KL) divergence. Further, they express this function as a dot product between query and passage representations, thereby allowing for efficient retrieval by means of standard MIPS (e.g., FAISS 70). Finally, they report state-of-the-art retrieval performance and show that the predicted covariance matrix can be used as a pre-retrieval query performance predictor.

Even though the results reported in the original study showcase the effectiveness of the proposed method, our study is motivated by several important questions that still need to be explored. Many of these questions have the character of a reproducibility question. First and foremost, the unavailability of the source code and model checkpoints makes it difficult to verify the paper's substantial claims. Second, in the original work, a batch of 512 was used to train MRL, indicating that substantial computational resources were available–in contrast, some of the baselines were trained with a batch size of 8 on a single GPU. Third, even though the model consists of various components and several stages of knowledge distillation, the original work does not include an extensive ablation study that explores the impact of each component on downstream performance. Therefore, it is unclear to what degree the performance gains of the proposed method stem from the multivariate representations of queries and passages. To this extent, it is important to understand the representations learned by MRL. In this chapter, we aim to answer **RQ3**. To do so, we break it down into the following research sub-questions:

RQ3.1 To what extent can the results and findings of the original paper be reproduced under a memory-limited setup?

RQ3.2 Do the multivariate query and passage representations express uncertainty?

RQ3.3 What is the contribution of each MRL component to the downstream retrieval performance?

RQ3.4 What is the impact of batch size in training MRL?

We summarize our contributions to the original work as follows:

Correction of a typographical/mathematical error We correct a mathematical error in the original work, made early in the formulation of the method, in an attempt by the authors to simplify the computation of KL divergence between two multivariate normal distributions. This error propagated to the rest of the original paper's mathematical formulations. We suspect that this error is typographical and that it did not leak into the technical implementation of the original authors' experiments. In fact, we provide experimental evidence that if the incorrect similarity scoring function is used instead of our corrected version, it harms retrieval performance.

Reproducing retrieval and QPP experiments We reproduce the experimental setup of the original paper, for the tasks of dense retrieval and pre-retrieval query performance prediction (QPP). The original MRL model is trained with a batch size of 512, while its primary competing approach uses a batch size of 8. We explore to what extent the original work's findings are reproducible when MRL is trained with a significantly smaller batch size under a memory-limited setup that facilitates fair comparisons of MRL against its baselines (i.e., both MRL and its baselines are trained with the same batch size). To answer RQ3.1, we focus on reproducing the retrieval experiments of the original work, in a memory-limited setup. We showcase that even though MRL can still achieve state-of-the-art retrieval results, retrieval performance significantly drops under a memory-limited setup. Moreover, we show that MRL does not outperform

the baselines in a fair comparison. Finally, regarding RQ3.2, MRL yields inconsistent results for the QPP experiments. Our analysis reveals that the variance vectors do not consistently capture notions of uncertainty.

Ablation study on the model's components The MRL framework is composed of multiple components, including multivariate representations, knowledge distillation, and model initialization from an already effective pre-trained dense retriever. To answer RQ3.3, we conduct a thorough ablation study on the model's components to unveil their importance in training an effective retriever. We find that multivariate representations do not boost the model's performance, and the high effectiveness stems from the model initialization and knowledge distillation from the re-ranker.

Ablation study on the batch size In the original paper, MRL performance is reported only for training with a batch size of 512. Therefore, it remains to be seen how the batch size affects performance. We conduct a thorough ablation study on the batch size to unveil its effect on the retrieval performance of MRL (RQ3.4). We believe that the impressive results reported in the original paper are due to training with a large batch size for many training steps.

Proposed improvements upon the original MRL model We propose a simple alteration to the original model, which results in a reduced hyperparameter search space. In short, instead of a parametric softplus activation which ensures positive semi-definiteness of the covariance matrix, we propose predicting the log-variance instead, which obviates searching for the β hyperparameter of the softplus function. We show that the log-variance model either matches or outperforms the original softplus model.

4.2 Related Work

Uncertainty-aware retrieval Uncertainty estimation in neural information retrieval (IR) has been explored in the past, although not in the context of dense (first-stage) passage retrieval, which is the main novelty aspect of the MRL method. The work of Cohen et al. [25] and Heuss et al. [61] focuses on risk-aware (second-stage) re-ranking. Both approaches attempt to approximate Bayesian models that predict a distribution of relevance scores rather than point estimates; the former uses Monte-Carlo dropout [42], while the latter leverages Laplace approximation. The common denominator across these Bayesian methods is that the predictive distribution $p(y|\theta, \mathcal{D})$ is approximated by performing forward inference using multiple samples of θ. This is the main difference between prior work and MRL: In MRL, predictive uncertainty is not framed as weight uncertainty, and variance does not represent deviation from the mean prediction. Rather, variance in MRL is a predicted value of a deterministic estimator.

Uncertainty for detecting out-of-distribution/corruptions While a variety of efforts exist in the area of stochastic representations in image retrieval [21, 167], recent work by Warburg et al. [169] showed that Bayesian image retrieval with Laplace approximation can achieve some desirable properties. They show that the uncertainty of prediction

increases (almost monotonically) with the amount of corruption in the input. The model's predictive uncertainty further behaves as expected when making out-of-domain predictions. These insights are valuable in text retrieval as well, where these desirable properties have not yet been achieved effectively.

Knowledge distillation The MRL framework is surrounded by multiple layers of knowledge/parameter distillation, which we summarize in this section. First, the selected neural architecture employed in MRL is DistilBERT [133]; a distilled version of BERT with 40% fewer parameters for 97% of its original performance. Furthermore, the architecture has been distilled with balanced topic-aware sampling (TAS-B) [64], which uses two teacher models to construct better training batches. Finally, MRL itself uses a knowledge distillation loss inspired by CLDRD [188]. While the original paper does not discuss how these sources of distilled knowledge affect downstream performance, in Section 4.4 of this chapter we perform a thorough ablation study that examines them one-by-one.

4.3 Methodology

The proposed MRL framework represents queries and passages as multivariate Gaussian distributions. It does so by computing a mean vector μ and a diagonal covariance matrix Σ for a query q and passage d, using query and passage encoders f_θ and f_ϕ, parameterized by θ and ϕ respectively,

$$(\mu_Q, \Sigma_Q) = f_\theta(q), \tag{4.1}$$

$$(\mu_D, \Sigma_D) = f_\phi(d). \tag{4.2}$$

It is also possible to have $\theta = \phi$, i.e., weight sharing, which the authors of the original paper opt for. Dense retrieval models (e.g., DPR, 73) typically use the embedding of a special token, the `[CLS]` token as the low-dimensional representation of queries and documents. Given a piece of text, for instance, "Hello world", pre-processing appends the special `[CLS]` token to the start of the text, producing "`[CLS]` Hello world", and the output of the transformer model for the `[CLS]` token is used. Relevance is a function of query and document representations, typically a dot product or cosine similarity. MRL, however, produces two vectors per input, which motivates the choice in the original study to use an additional special token, termed the `[VAR]` token, appended after the `[CLS]`token, but before the text. For instance, "Hello world" is pre-processed to "`[CLS] [VAR]` Hello world". The output representation of the `[CLS]`token is used to compute the mean, and the output representation of the `[VAR]` token is used to compute the variance.

The relevance score between queries and passages is then defined as the negative KL divergence between their distributional embeddings: $Q \sim \mathcal{N}(\mu_Q, \Sigma_Q)$ and $D \sim \mathcal{N}(\mu_D, \Sigma_D)$,

$$\text{rel}(q, d) = -\,\text{KLD}(Q \mid D). \tag{4.3}$$

The minus sign is there to implement a "higher is better" type of scoring. To simplify matters, we will disregard it in the upcoming derivations and re-introduce it at the very

end. In this section, we detail the reproducibility study of the above framework. First, we direct attention to a small mathematical error that was made early in the formulation of the relevance scoring in the original paper, that propagated through the rest of the mathematical derivations. We then discuss matters of model training. Whenever we make a strong assumption due to the lack of implementation detail in the original paper, or the lack of shared source code, it is explicitly mentioned.

4.3.1 KL divergence-based relevance scoring

In Eq. 4.4, we start by repeating the standard definition of KL divergence, as written in Eq. 9 of the original paper:

$$\begin{aligned}\mathrm{KLD}(Q \mid D) = \frac{1}{2}\Big[\log \frac{\det \Sigma_D}{\det \Sigma_Q} - k + \mathrm{tr}\{\Sigma_D^{-1}\,\Sigma_Q\} \\ + (\mu_Q - \mu_D)^\intercal\, \Sigma_D^{-1} (\mu_Q - \mu_D)\Big],\end{aligned} \tag{4.4}$$

where k denotes the dimensionality of the multivariate Gaussian embeddings. For the purpose of relevance scoring, the authors proceed to further simplify Eq. 4.4 and reformulate it as a document ranking function. They do so by eliminating document-independent terms and constants, and by taking advantage of the fact that the covariance matrices are diagonal. Let us follow their simplification steps by considering each term separately. For the first term we have,

$$\begin{aligned}\log \frac{\det \Sigma_D}{\det \Sigma_Q} = \log \det \Sigma_D - \underbrace{\log \det \Sigma_Q}_{\substack{\text{constant w.r.t.} \\ \text{doc. ranking}}} = \log \det \Sigma_D \\ = \log \prod_{i=1}^{k} \sigma_{i_D}^2 = \sum_{i=1}^{k} \log \sigma_{i_D}^2.\end{aligned} \tag{4.5}$$

The subsequent steps in the original paper contain an error in the simplification of the second term. We include the original formulation in Appendix 4.A.1 for completeness. We note that using the original formulation leads to drastically lower performance, which makes it likely that this error is typographical i.e., it did not propagate to the implementation (see Section 4.5.1 for more details). We provide the correct derivation in Eq. 4.6 as follows:

$$\mathrm{tr}\{\Sigma_D^{-1}\,\Sigma_Q\} = \sum_{i=1}^{k} \frac{\sigma_{i_Q}^2}{\sigma_{i_D}^2}. \tag{4.6}$$

Finally, for the third term we have,

$$\begin{aligned}(\mu_Q - \mu_D)^\intercal\, \Sigma_D^{-1} (\mu_Q - \mu_D) = \sum_{i=1}^{k} \frac{(\mu_{i_Q} - \mu_{i_D})^2}{\sigma_{i_D}^2} \\ = \sum_{i=1}^{k} \frac{\mu_{i_Q}^2}{\sigma_{i_D}^2} - \sum_{i=1}^{k} \frac{2\mu_{i_Q}\mu_{i_D}}{\sigma_{i_D}^2} + \sum_{i=1}^{k} \frac{\mu_{i_D}^2}{\sigma_{i_D}^2}.\end{aligned} \tag{4.7}$$

Combining Eq. 4.5, 4.6 and 4.7 into Eq. 4.4, and removing constants, we arrive at the intended derivation of the ranking function:

$$\mathrm{KLD}(Q \mid D) = \sum_{i=1}^{k} \log \sigma_{i_D}^2 + \sum_{i=1}^{k} \frac{\sigma_{i_Q}^2}{\sigma_{i_D}^2} + \sum_{i=1}^{k} \frac{\mu_{i_Q}^2}{\sigma_{i_D}^2} - \sum_{i=1}^{k} \frac{2\mu_{i_Q}\mu_{i_D}}{\sigma_{i_D}^2} + \sum_{i=1}^{k} \frac{\mu_{i_D}^2}{\sigma_{i_D}^2}. \quad (4.8)$$

Note that unlike Eq. 4.4, Eq. 4.8 is no longer the KL divergence. After all the simplifications, it is a KL divergence-based relevance scoring function for *ranking* documents given a query. From this point forward, we continue with the work described in the original paper, but we base it on our Eq. 4.8, which is the derivation of the relevance scoring function that includes our correction.

The next step of this reproducibility study is to express Eq. 4.8 as a dot product between query and passage vectors i.e., $\mathrm{KLD}(Q \mid D) = q^\intercal \cdot d$, with the purpose of reusing standard efficient inner product similarity search [70]. To do so, we isolate the document-specific terms of Eq. 4.8 that can be pre-computed:

$$\gamma_D = \sum_{i=1}^{k} \left(\log \sigma_{i_D}^2 + \frac{\mu_{i_D}^2}{\sigma_{i_D}^2} \right). \quad (4.9)$$

In the original paper, the term γ_D is referred to as a "document prior". Now we can express the relevance score as a dot product between query and passage vector representations:

$$\vec{q} = \left[1, \sigma_{1_Q}^2, \ldots, \sigma_{k_Q}^2, \mu_{1_Q}^2, \ldots, \mu_{k_Q}^2, \mu_{1_Q}, \ldots, \mu_{k_Q} \right], \quad (4.10)$$

$$\vec{d} = \left[\gamma_D, \frac{1}{\sigma_{1_D}^2}, \ldots, \frac{1}{\sigma_{k_D}^2}, \frac{1}{\sigma_{1_D}^2}, \ldots, \frac{1}{\sigma_{k_D}^2}, -\frac{2\mu_{1_D}}{\sigma_{1_D}^2}, \ldots, -\frac{2\mu_{k_D}}{\sigma_{k_D}^2} \right], \quad (4.11)$$

where $\vec{q}, \vec{d} \in \mathbb{R}^{1\times(3k+1)}$. At this point, we remind the reader that following Eq. 4.3, the relevance score is the negative of the KL divergence.

4.3.2 Listwise knowledge distillation

Knowledge distillation has become of great importance in boosting the effectiveness of dense retrievers [63]. In detail, a highly effective cross-encoder re-ranker is used as a teacher to transfer knowledge to a less effective but efficient first-stage dense retriever student model. Consequently, the effectiveness of the dense retriever is increased while it retains its efficiency. In the original work by Zamani and Bendersky [187], the authors employ a listwise distillation loss function [188] to train their dense retriever (i.e., student model). For each query q and a set of passages D_q (later in this section we provide details on how this set is constructed), the loss is computed as:

$$\sum_{d,d' \in D_q} \mathbb{1}\{y_q^t(d) > y_q^t(d')\} \left| \frac{1}{\pi_q(d)} - \frac{1}{\pi_q(d')} \right| \log(1 + e^{M_\theta(q,d') - M_\theta(q,d)}), \quad (4.12)$$

where $\pi_q(d)$ denotes the position of passage d in the ranked list produced by the dense retrieval student model M_θ and $y_q^t(d)$ denotes the relevance judgment produced by the

teacher model for the pair of query q and passage d; $y_q^t(d)$ can be either a score or a label. In the original work of Zamani and Bendersky [187], $y_q^t(d)$ is the raw score from the teacher model for a query-passage pair, and D_q is constructed as follows. Given a query q, the passage set D_q is constructed with positive passages provided by the dataset's official relevance judgments. On the other hand, the negative passages are sampled from the top-k passages retrieved with BM25 and the top-k passages retrieved by the student dense retrieval model itself. Finally, the original work uses in-batch negative training to reuse passages from other queries that are already in the batch.

4.4 Experimental Setup

4.4.1 Datasets and metrics

Our evaluation is performed on both in-domain (ID) and out-of-domain (OOD) data. In the OOD setting, we perform zero-shot evaluation. All tasks are ad-hoc retrieval, with a fixed set of documents. Statistics of the datasets are reported in Appendix 4.A.4. We summarize the datasets and evaluation methodology below.

In Domain (ID) We train all models on the MS-MARCO [113] training set. Note that we split the full training set into a training and validation set for hyperparameter tuning as described in Section 4.4.4. There are three in-domain evaluation sets, all of which are based on the MS-MARCO corpus. This includes the MS-MARCO Dev set, the TREC-DL 2019 [27] and TREC-DL 2020 [28] datasets. Both TREC datasets are densely labeled by humans. The evaluation metric for the Dev set is the mean reciprocal rank (MRR) with a cut-off of 10, denoted as MRR@10. For the TREC subsets, we use the standard evaluation metrics of normalized discounted cumulative gain at 10 (nDCG@10), and mean average precision (MAP).

Out of Domain (OOD) We evaluate the retrieval models' generalization ability in different domains via zero-shot passage retrieval experimentation. All retrieval models are trained on the MS-MARCO training set and tested on previously unseen queries and underlying corpus. We replicate the evaluation setup outlined in [187], with nDCG@10 as the primary metric. We evaluate the following OOD datasets in zero-shot setting: (i) SciFact [164]: a scientific claim verification dataset where the task involves retrieving abstracts that either refute or support a claim, (ii) FiQA [97]: a dataset that involves retrieval of documents in the financial domain using natural language questions, (iii) TREC-COVID [161]: a biomedical dataset of scientific articles about COVID-19, with questions as the topics/queries, and (iv) CQADupStack [65]: a community question answering (CQA) dataset, with the task of retrieving duplicate questions in a community website (StackOverflow).

4.4.2 Baselines

We compare MRL against the following single-vector dense retrieval models:

- **DPR** [73]: is a traditional dense retriever that is trained with softmax cross-entropy.

- **TAS-B** [64]: is an effective dense retriever that is trained by combining (i) knowledge distillation from a re-ranker teacher model (i.e., cross-encoder) with (ii) a balanced topic-aware sampling method. This method alternates the creation process of the training batches by composing batches based on queries clustered in the same topic. Furthermore, it selects passage w.r.t. the pairwise margin between positive and negative passages in the batch so that the margin of positive-negative pairs is balanced in the margin range.
- **CLDRD** [188]: is a state-of-the-art dense retriever that uses TAS-B as initialization and is trained by combining curriculum learning with knowledge distillation; in particular, it uses the listwise loss of Eq. 4.12. The student dense retriever is trained via an iterative training process in which the difficulty of the training data, produced by the re-ranking teacher model, increases with each iteration. Additional information regarding the training setup of CLDRD can be found in the Appendix 4.A.3.

The motivation behind selecting these baselines is twofold: First, their inclusion in the original study, and second, to enable fair comparisons in our subsequent ablation study. For instance, MRL can be compared with CLDRD to assess the impact of the multivariate representations, and a similar assessment can be made when MRL without distillation is compared against DPR.

4.4.3 Query performance prediction

The QPP task [15, 56, 57] involves inferring the difficulty of a given query for a search system without using relevance judgments. We replicate the pre-retrieval QPP setup in [187], evaluating on the TREC-DL 19 and TREC-DL 20 datasets. That is, we retrieve documents for a given query using a search system and evaluate it using nDCG@10 to obtain a ground-truth assessment of performance. Then, we use a QPP method to predict the performance and evaluate it against the ground-truth assessment using three correlation measures, Spearman's correlation, Pearson correlation, and Kendall's Tau.

The effectiveness of a QPP method is a function of the underlying retrieval system. Since it was unclear from the original study which system was used to compute the ground truth performance, we experiment with multiple search systems.

In addition to the model itself, we experiment with three retrieval models independent of the MRL model, to measure how well the QPP method generalizes. We include a traditional lexical retriever (BM25), a simple dense retriever (DPR), and an effective dense retriever (TAS-B). We use the following baselines used in the original study:

- **SCQ** [193]: computes the similarity between a query and the corpus for each query term based on the frequency of occurrence of the term in the corpus.
- **VAR** [15]: considers the variance or standard-deviation of the term weights of each query term, based on the documents in which the term occurs.
- **IDF** [15]: is based on the inverse document frequency of each query term.
- **PMI** [55]: is a predictor that computes the pointwise mutual information, assigning high scores for frequently co-occurring query terms. Given all possible query term pairs, either the average or the maximum can be used as the predictor.

For SCQ, VAR, and IDF, the scores are computed at the query term level and then aggregated using either summing, averaging, or taking the maximum of each score. We report each of these aggregations in the results.

QPP for MRL Zamani and Bendersky [187] mention that the norm of the variance $|\Sigma_Q|$ is used to compute the predicted performance. We interpret this statement as using a *function* of the covariance, and use the negative norm $-|\Sigma_Q|$, because the predicted variance should *increase* for queries that are difficult, rather than decrease. For instance, a typographical error in the query makes it more difficult to address than a "clean" query [139, 140], leading to lower performance compared to a clean query. Similarly, an OOD query could also result in poorer performance on average compared to an ID query. From the QPP perspective, a model should therefore assign *lower* predicted performance for these types of queries, motivating our choice. In Section 4.5.2, we show that this intuition holds empirically.

4.4.4 Implementation details

We train MRL for $200K$ steps. In each step, we optimize the distillation loss (Eq. 4.12) using a batch of queries, one positive passage per query, and 30 negative passages per query; 5 of the negative passages are mined with BM25, and 25 are mined with the student model. With that setup, we use a batch size of 15 queries–the maximum that can fit in a 40GB A100 GPU, given the size of our model. We set the maximum length for queries and passages to 32 and 256 tokens, respectively. We initialize the dense retriever student model with the official TAS-B checkpoint, and we set as the teacher model the `ms-marco-MiniLM-L-6-v2` cross-encoder that is publicly available on HuggingFace. We use an Adam optimizer with a learning rate of 5×10^{-6}, and linear learning rate scheduling with warm-up for 10% of the training steps. The β parameter for softplus is set to 2.5. For MRL, the mean and variance are obtained by passing the `[CLS]` token and a `[VAR]` token respectively through fully connected projection layers. The MRL models reported use means and variances projected down to 383 ($= \frac{768}{2} - 1$).

Since MS-MARCO does not include a validation set, we split the training set into a validation (6890 queries) and a training set. The parameters above were selected after a hyperparameter search with the validation set performance used to pick the best model. Refer to Appendix 4.A.5 for the full set of hyperparameters. We use the `Tevatron` toolkit [46] to train the models and the `pytrec_eval` library [159] to evaluate the retrieval performance. Finally, our QPP baselines are based on an existing implementation by Meng et al. [103].

4.5 Discussion

We organize the discussion section around retrieval experiments in Section 4.5.1, the investigation of the variance vectors in Section 4.5.2, and the results of the ablation study in Section 4.5.3.

Table 4.1: Reproduction results of MRL on our memory-limited setup. The upper part contains the reproduction results, while the lower part contains the results reported in the original study.

	Model	MS MARCO		TREC-DL'19		TREC-DL'20	
		MRR@10	MAP	NDCG@10	MAP	NDCG@10	MAP
Reproduced	DPR	.312	.319	.649	.345	.625	.356
	TAS-B	.344	.351	.721	.396	**.685**	.430
	CLDRD	**.378**	**.383**	**.727**	.448	.670	.446
	MRL	.373	.378	.707	**.462**	.681	**.461**
Reported	TAS-B	.344	.351	.717	.447	.685	.455
	CLDRD	.382	.386	.725	.453	.687	.465
	MRL	.393	.402	.738	.472	.701	.479

4.5.1 Reproducing the retrieval results

In this study, we work under a memory-limited setup–an academic computational budget with constrained access to a single GPU. We start by testing whether we can obtain the results reported in the original study under these memory constraints. We report the results in Table 4.1 and Table 4.2, where DPR, CLDRD, and MRL are our implementations of the original methods. We report results for TAS-B by using the official pre-trained checkpoint, which also serves as CLDRD and MRL initialization. This way, we can ensure a fair comparison between the different methods. We include the original numbers in the lower group in Tables 4.1 and 4.2. At this point, we want to underline that for MRL we use the corrected KL formulation we presented in Section 4.3 for our experiments. Our decision to do so is grounded in the belief that the original study's authors also used this formulation in their implementation and that the formulation with the mathematical error is a typographical mistake in their paper. We arrived at this conclusion based on our preliminary experiments, which yielded a dramatically low retrieval performance (i.e., MRR@10 was 0.229 for MS-MARCO) when following the wrong formulation.

We first focus on testing whether we are able to replicate the results for CLDRD, the main competitor of MRL. Our experimental results affirm the state-of-the-art retrieval performance (for single-vector dense retrievers) of CLDRD. Furthermore, our findings validate the original study regarding its ability to enhance the performance of TAS-B, both in the ID (see Table 4.1) and OOD (see Table 4.2) scenarios. We consider this a successful replication despite the slight discrepancy in the results. The reason for not obtaining the exact same results can be attributed to different development toolkits, hardware, implementation details not present in the original work, etc.

Regarding the reproduction of MRL, we were unable to yield the same results as the original study. We observe a drop in performance across all reported metrics for both the ID and OOD datasets. For instance, in the case of FiQA, our implementation

Table 4.2: Reproduction results of MRL on our memory-limited setup for the zero-shot retrieval settings. The upper part contains the reproduction results, while the lower part contains the results reported in the original study.

	Model	SciFact NDCG@10	FiQA NDCG@10	TREC-COVID NDCG@10	CQADupStack NDCG@10
Reproduced	DPR	.474	.231	.600	.266
	TAS-B	**.643**	.301	.481	.313
	CLDRD	.627	**.308**	**.608**	**.327**
	MRL	.591	.291	.497	**.327**
Reported	TAS-B	.643	.300	.481	.314
	CLDRD	.637	.348	.571	.327
	MRL	.683	.371	.668	.341

yielded an NDCG@10 of 0.308, which is lower than the original study's reported value of 0.371. Furthermore, we could not find the trends that were reported in the original study. Specifically, the original study showed that MRL outperformed both TAS-B and CLDRD in ID and OOD scenarios. In contrast, in our work, MRL achieves similar performance to CLDRD on the ID datasets. A similar trend holds for the OOD dataset, except for TREC-COVID, where CLDRD outperforms MRL with a substantially higher score of 0.608 compared to 0.481 for MRL. Upon comparing MRL with TAS-B, it is noted that MRL either matches or outperforms TAS-B in the ID datasets, except for TREC-DL 20. However, in the OOD datasets, MRL surpasses TAS-B only for TREC-COVID and CQADupStack.

We stress that our goal in this work is not to replicate MRL; an exact replication is not possible since the original manuscript does not specify several design choices–the best hyperparameters (i.e., learning rate, *softplus* β, number of training steps), the cross-encoder model that was used as a teacher, as well as the number of negative passages per query in a batch. We aim to reproduce MRL under our memory-limited setup, which facilitates fair comparisons against the baselines. From our experimental results, we conclude that although MRL is a competitive approach in our setup, the multivariate representations do not boost the retrieval performance. Furthermore, we unveil that MRL cannot consistently outperform its competitors when evaluated under fair comparisons. However, different from CLDRD, MRL produces a variance that can be used in downstream tasks. We investigate the utility of this predicted variance in the following section.

4.5.2 Analyzing the variance

We analyze the predicted variance in three experiments: query performance prediction experiments (Section 4.5.2), experiments with typos, and retrieval experiments with alternative encoding schemes in Section 4.5.2. The first is a replication of the QPP experiments included in the original paper, while the latter two are additional analyses.

Table 4.3: QPP results for MRL for four reference models. While MRL does perform well for TREC-DL 20 for BM25, it fails to do so for TREC-DL 19. The opposite is true for DPR. For both TAS-B and MRL reference models, MRL is more consistent but fails to reach the reported performance (bottom row). Furthermore, as is evident from Figures 4.1 and 4.2, MRL is outperformed by simple baselines in most comparisons.

		TREC-DL 19			TREC-DL 20		
		S-ρ	P-ρ	K-τ	S-ρ	P-ρ	K-τ
MRL	BM25	-.029	-.034	-.014	.275	.296	.190
	DPR	.233	.171	.155	-.051	-.013	-.041
	TAS-B	.204	.196	.138	.177	.166	.117
	MRL	.171	.182	.134	.157	.185	.106
	MRL (reported)	-	.271	.259	-	.272	.298

Query performance prediction

The results for the QPP experiments are plotted in Figures 4.1 and 4.2. We also report the MRL results separately in Table 4.3, which also includes the reported numbers in [187]. As mentioned previously, we used four reference models because the original study did not report which model was used, and also to see if MRL generalizes to different reference models. We note that the original paper does not mention how the norm is used in computing the predicted performance–in our experiments, we use the *negative* norm (using the norm flips the signs of the correlation)–intuitively, a higher uncertainty should result in lower performance (see Section 4.4.3).

From Table 4.3, we were unable to reproduce the numbers for MRL reported in the original paper, with any of the reference models. While MRL achieves higher than reported numbers for TREC-DL 20 with BM25, this result does not hold for TREC-DL 19, where there appears to be a random correlation. This trend is flipped with DPR as the reference–MRL performs well for TREC-DL 19 but not TREC-DL 20. For TAS-B and DPR, MRL is more consistent. However, the correlation obtained is lower than in the original study. But, how does MRL compare with the baselines?

Figure 4.1 contains results for TREC-DL 19, with each subplot corresponding to the three metrics we used. Comparing MRL (top row) with the other methods, we notice that at least one baseline outperforms MRL for each metric regardless of the reference model. In particular, at least one variant of the PMI baseline outperforms MRL. We remind the reader that these baselines are simple, non-parametric methods that use statistics derived from the corpora to compute the query difficulty.

For TREC-DL 20, the results are more encouraging. If the reference model is MRL itself, we observe that MRL beats every other baseline for all three metrics. However, this result does not generalize to the other three reference models, where similar trends to TREC-DL 19 are observed, with at least one baseline (among PMI, VAR, IDF) beating MRL. We observe similar trends for three reference models BM25, TAS-B, and DPR, with MRL outperformed by at least one baseline.

In these experiments, we investigate the degree to which the MRL framework

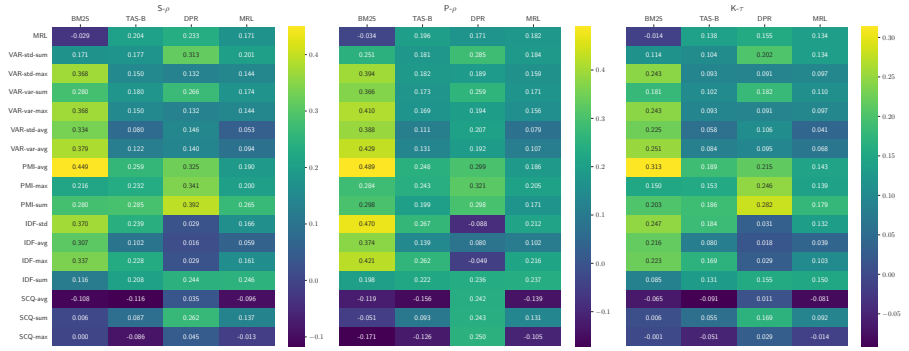


Figure 4.1: QPP results for TREC-DL 19 for four reference models (x-axis) and several methods (y-axis). Each subplot corresponds to a different correlation metric: Spearman's (S-ρ), Pearson's (P-ρ), and Kendall's Tau (K-τ) correlations. MRL is outperformed by simple baselines regardless of the reference model or metric.

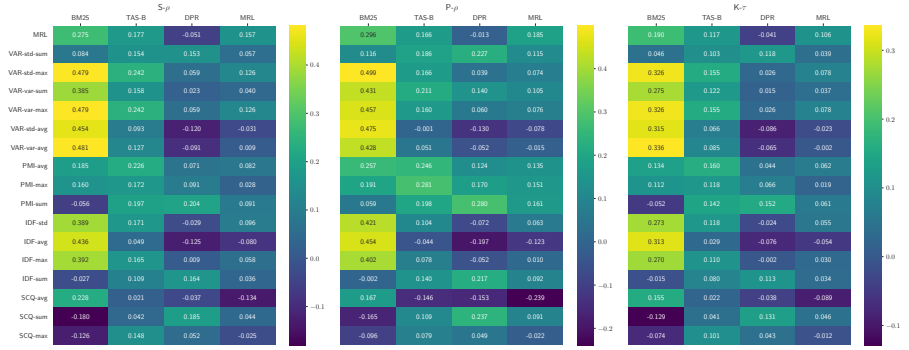


Figure 4.2: QPP results for TREC-DL 20. Simple lexical baselines outperform MRL when the reference is BM25, TAS-B, or DPR. With MRL itself as the reference, MRL achieves better performance, often by large margins.

captures the notion of uncertainty by measuring a proxy–query difficulty. Intuitively, this suggests that for some datasets and reference models, higher uncertainty was indeed assigned to difficult queries. However, the positive correlation is *not consistent* across datasets. MRL fails to generalize to different reference models and datasets, achieving random correlation in many settings. Furthermore, MRL is outperformed by simple non-parametric baselines in most comparisons. The lack of consistent and strong correlations suggests that MRL is unlikely to be a strong and consistent predictor of query difficulty in our experimental setup. Motivated by these results, we explore what the predicted variance captures in the next section.

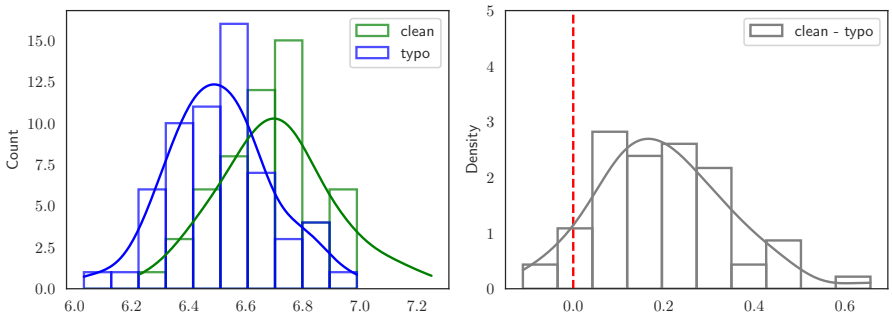


Figure 4.3: Uncertainty of clean and corrupted data: We plot the distributions of the norm of the predicted variance of the clean and corrupted data on the left. MRL assigns *lower* uncertainty to the corrupted data compared to the clean data. On the right, we plot the distribution of $|\Sigma_{\text{clean}}| - |\Sigma_{\text{corrupted}}|$, which is mostly positive (i.e., right of the red-dotted line). Contrary to expectations, MRL fails to assign higher uncertainty to corrupted data.

Does MRL capture uncertainty?

We outline two additional experiments investigating the predicted variance beyond the original paper: (a) contrasting the predicted variance of corrupted and clean data and (b) experimenting with alternate encoding schemes.

Experiments with typographical errors Here, we consider an analog to the QPP experiments above, but instead of retrieval difficulty, we examine if the model is sensitive to *data distribution shifts* instead. Inspired by works in the vision domain (e.g., 167, 169), we argue that a model should assign higher variance to *corrupted* or *OOD data* compared to clean or ID data.

We experiment with the DL-Typo [197] dataset that contains 60 query pairs accompanied by relevance assessments. Each pair consists of a real user query with typographical errors and its corresponding version where these errors have been corrected. For instance, the corrupted query "what is acid reflex" and its typo-free version "what is acid reflux". Given these data, we compute the norms of the predicted variance and plot their distributions. Suppose the model indeed models uncertainty accurately. In that case, we expect (a) clean data should be assigned lower variance compared to corrupted data, (b) the distributions of the clean and corrupted data are well separated, and (c) the differences between the clean and corrupted data, i.e., $|\Sigma_{\text{clean}}| - |\Sigma_{\text{corrupted}}|$, should be negative.

We plot these distributions in Figure 4.3. While we expected more variance to be assigned to the corrupted data, the opposite is true. There is some separation observed between the two distributions–the distribution of the corrupted data ("typo") is to the left of the clean data. The right plot underscores this result, as the $|\Sigma_{\text{clean}}| - |\Sigma_{\text{corrupted}}|$ distribution is mostly positive. This suggests that contrary to expectation, the model predicts a higher variance for clean data. We expand on this analysis by examining the

Table 4.4: Performance of different encoding schemes for the MRL model. The first row uses the original equations proposed in the original paper, whereas the second row includes our corrections. The retrieval performance is measured with MRR@10 for MS MARCO and NDCG@10 for the rest of the datasets.

Encoding	MS MARCO	TREC-DL'19	TREC-DL'20	SciFact	FiQA	TREC-COVID	CQADupStack
Eq. 4.15 & 4.16 [187]	.229	.525	.486	.096	.116	.215	.183
Eq. 4.10 & 4.11 (Ours)	.373	.707	.681	.591	.291	.497	.327
Mean	.279	.624	.557	.382	.168	.145	.187

role of the predicted variance in retrieval relevance.

Encoding MRL produces both a mean and variance for queries and documents. In Table 4.1 and Table 4.2, we reported retrieval results using the encoding scheme which enables retrieval using the KL divergence, i.e., documents are encoded and indexed with Eq. 4.11 and queries with Eq. 4.10. If the variance *only* captures uncertainty, we argue that the difference in retrieval performance when *only* the mean is used should not be much lower than the performance obtained with this encoding scheme. However, as Table 4.4 shows, this is not true. Comparing the encoding scheme (second row) with using just the mean (third row), we see that performance drops sharply across all datasets, especially for the OOD datasets.

The performance drop could be explained partly due to the way the models were trained. Since the full KL loss (Eq. 4.4) was used in training the model, it may not be equipped to perform retrieval using just the mean. However, intuitively, we expect the mean to model *relevance* and variance to model *uncertainty*, which means that the retrieval performance should not be drastically different when only the mean is used for retrieval. The drastic drop suggests that the model may be instead using the predicted variance vectors as a signal for relevance. This is the core difference between a method such as MRL and a Bayesian method: variance in MRL is not statistical variance, i.e., it does not express deviation from the mean prediction. Instead, variance in MRL is a deterministically estimated quantity that minimizes a distance objective.

In this section, we examined if the MRL model consistently predicts a notion of variance that reflects a notion of uncertainty defined by either query difficulty or sensitivity to data distribution shifts. We find that the QPP results are inconsistent, and against initial expectations, the model assigns a higher uncertainty to corrupted data. In addition, experiments with encoding using only the mean suggest that the variance seems to model relevance since performance drops drastically when only the mean is used for retrieval.

4.5.3 Ablation study

Even though the MRL performance reported in the original paper was not reproduced under our experimental setup, MRL remains a framework that can produce a highly effective dense retriever even under memory constraints. MRL consists of several components: multivariate representations, knowledge distillation, model initialization from a pre-trained dense retriever, and a batch construction strategy. Thus, it needs to be made clear how much each component impacts effectiveness. We expand on the original paper's findings by studying each component's contribution to retrieval performance through experimentation with various MRL variants.

Multivariate representations MRL represents queries and passages as multivariate distributions and uses negative multivariate KL divergence to compute similarity. First and foremost, we want to understand how much the multivariate representations contribute to the overall effectiveness of the retriever. In order to test this, we conduct an experiment where we substitute multivariate representations with vector representations and compute similarity via the dot product. We find that multivariate representations do not lead to a higher retrieval performance; in contrast, we observe a small decrease in performance when used (see rows 1 and 2 in Table 4.5).

Model initialization Since MRL's initialization point is a pre-trained, highly effective dense retriever, namely TAS-B, it is important to examine the impact of its initialization on its downstream performance. To test this, we use DistilBERT as the initialization instead. When comparing row 1 against row 3 in Table 4.5, we see that it is possible to train a competitive model with a DistilBERT initialization; however, it cannot achieve state-of-the-art performance.

Loss function Besides training with knowledge distillation, MRL can be trained with supervised contrastive learning by minimizing a softmax cross-entropy loss [73]. In row 4 of Table 4.5, we test this alternative and find that following a knowledge distillation training scheme is crucial for training an effective retriever. This result is in line with recent works in dense retrieval that have shown the superiority of knowledge distillation training over supervised contrastive learning [64, 91].

Negatives from the student model The MRL framework uses the student model to sample hard negatives (see Section 4.3.2). This can be achieved by using a recent checkpoint from the student model and updating an ANN index of the corpus representation used for negative sampling. The encodings of the documents in the corpus can be updated multiple times during training, allowing us to sample new hard negatives from the ANN index more than once. In our reproducibility work, we update the ANN index only once. However, we also experiment with updating the ANN index multiple times (i.e., every 75k steps). In our preliminary experiments, updating and sampling from the ANN index more than once does not significantly increase performance: $\mathrm{MRR@10} = .374$ for MS MARCO, $\mathrm{NDCG@10} = .714$ for TREC-DL'19, and $\mathrm{NDCG@10} = .685$ for TREC-DL'20 (these scores can be compared against the first row in Table 4.5). We believe this can be attributed to the usage of TAS-B as initialization for the student

model, and as a result, high-quality hard negatives are available even from the first update.

Training strategy CLDRD, which is the primary competing approach of MRL, uses a different training setup when training with listwise knowledge distillation. In particular, CLDRD constructs the training batch using the teacher model to produce the data, the teacher's relevance judgments have the form of pseudo-labels (w.r.t. the controlling term $\mathbb{1}\{y_q^t(d) > y_q^t(d')\}$ in Eq. 4.12), it does not use in-batch negatives, and it uses curriculum learning (see Section 4.4.2 and Appendix 4.A.3 for more details). In contrast, MRL constructs the training batch via the ground-truth query relevance judgments (i.e., qrel file) and negative mining, and the teacher's relevance judgments are the raw scores from the model. We proceed with using CLDRD's training setup to train MRL; we refer to this model variant as MRL-CLDRD. This approach ensures a fair comparison with CLDRD and allows us to attribute any performance increase solely to MRL's multivariate representations. When we compare rows 1 and 5 in Table 4.5, we observe that MRL can slightly benefit from incorporating the training setup of CLDRD. Furthermore, by comparing rows 5 and 6, we notice that the multivariate representations cause a slight decrease in performance, consistent with the previous result.

Table 4.5: Ablation study on the different components of the MRL framework. The first part of the table examines the influence of initialization (i.e., TAS-B), loss function (i.e., Listwise KD), and multivariate representations for queries and passages on MRL performance. The second part explores the impact of training MRL using the curriculum learning framework of CLDRD on performance.

#	Variant	MS MARCO MRR@10	TREC-DL'19 NDCG@10	TREC-DL'20 NDCG@10	SciFact NDCG@10	FiQA NDCG@10	TREC-COVID NDCG@10	CQADupStack NDCG@10
1	MRL Multivariate representation TAS-B Listwise KD Teacher raw scores Qrels & Negative Mining	.373	.707	.681	.591	.291	.497	.327
2	- Multivariate representation + Vector representation	.375	.719	.686	.637	.306	.618	.334
3	- TAS-B + DistilBERT	.356	.693	.677	.567	.264	.536	.320
4	- Listwise KD + Cross-entropy	.328	.629	.644	.498	.245	.473	.268
5	MRL-CLDRD - Teacher raw scores - Qrels & negative mining + Teacher constructs the batch + Teacher pseudolabels	.375	.721	.667	.605	.293	.510	.320
6	CLDRD - Teacher raw scores - Qrels & negative mining - Multivariate representations + Teacher constructs the batch + Teacher pseudolabels + Vector representations	.378	.727	.670	.627	.308	.608	.327

4.5.4 The effect of batch size on MRL performance

The main goal of our study is to create a fair comparison of MRL against the baselines and explore to what extent the reported results of the original paper can still be achieved under a limited computational resource setting (i.e., an academic computation budget). Within our single GPU setting, we can train MRL with a batch size of 15. However, the original MRL work used a batch size of 512. At this point, we want to explore whether increasing the batch size can achieve the reported results of the original paper. Therefore, we experiment with training MRL for bigger batch sizes. The perks of such an experiment are two-fold: (i) we study the impact of batch size in the performance of MRL, and (ii) we unveil to what extent the reported results of the original work are due to the use of large batch size.

A batch larger than 15 does not fit into our available GPU. Fitting a batch of 512 would require approximately 34 A100 GPUs–something unfeasible to obtain with an academic computation budget. Hence, we adopt a memory reduction method to resolve the hardware bottleneck that is associated with training with batch-wise contrastive learning with a large number of negatives (even though with in-batch negatives, we can avoid encoding extra negatives, the loss of each sample is conditioned on every sample in the batch and therefore it is required to fit the entire large batch into GPU memory). Gao et al. [45] introduced a gradient caching technique (GradCache) that allows the production of the exact same gradient update as training with a large batch and experimentally showcased that it is possible to reproduce the state-of-the-art results of a dense retriever trained with a large batch on multiple professional GPUs in a single consumer-grade GPU.

We experiment with the following batch sizes: 15, 32, 64, 128, 256, and 512. We train all the models for the same number of epochs since there is significant additional training time due to GradCache's computational overhead. We experiment with MRL trained with in-batch negatives (as proposed in the original paper) and without in-batch negatives. As Figures 4.4 and 4.5 show, changes in batch size have a smaller impact on MRL when trained without in-batch negatives rather than with in-batch negatives. We hypothesize that the negative trend we witness as the batch size increases is linked to an increase in the number of (hard-negative, easy-negative) passage pairs contributing to the loss–where hard-negatives are passages sampled from BM25 and the student model while easy-negatives are the in-batch negatives. In detail, due to the use of in-batch negatives, as the batch size increases, there is an increase in (positive, easy-negative) passage pairs and an even greater increase in (hard-negative, easy-negative) pairs. The latter demands the dense retriever to recover finer-grained distinctions between the passages. Therefore, it is possible that as the batch increases and the number of such pairs increases, the training steps should also increase so that the model can learn such fine-grained differences.

To validate this, we set to train MRL with a batch size of 512 for longer to examine whether that could lead to better performance. As we can see in Table 4.6, training for longer can increase retrieval performance on all datasets. Thus, we hypothesize that the reported results in the original paper might be an outcome of training with 512 for a long time. Testing this hypothesis on our memory constraint environment, which consists of a single GPU and employs GradCache to simulate training with a large batch, is not

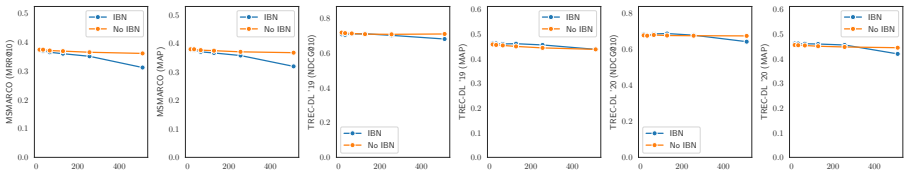


Figure 4.4: Impact of batch size on the MRL performance for ID datasets. IBN and No IBN indicate training MRL with in-batch and no in-batch negatives, respectively.

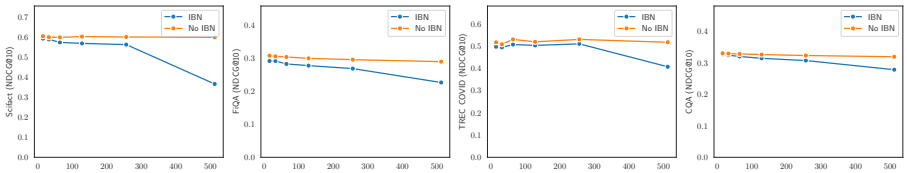


Figure 4.5: Impact of batch size on the MRL performance for OOD datasets. IBN and No IBN indicate training MRL with in-batch and no in-batch negatives, respectively.

feasible; as we show in Table 4.6, training MRL with batch size 512 on a single GPU with GradCache requires $\sim$8 days for 11K steps. At this point, we need to underline that such a result would suggest that the trends presented in the original paper may have been influenced by the fact that MRL was training with a significantly larger batch size compared to its baselines (i.e., in the original work, MRL is trained with a 512 batch, while CLDRD with a batch size of 8).

4.5.5 A simple extension to reduce the hyperparameter search space

The MRL model produces a mean and a diagonal co-variance matrix/variance vector given text input. Using the raw co-variance without ensuring that it is positive (and semi-definite) would make the model produce invalid values for the variance e.g., a negative variance. To ensure positivity, Zamani and Bendersky [187] pass the raw values through a *softplus* activation function (Softplus$(x) = \frac{1}{\beta} \cdot \log(1 + \exp(\beta \cdot x))$), which ensures that the predicted variance is positive. The original study finds that retrieval performance is robust to the hyperparameter β in the *softplus* function. An alternative to using the *softplus* function (that predicts a variance), is to predict the *log*-variance without any activation function. That is, we assume that the values produced by the encoder is the log-variance. As such, these values are exponentiated prior to use, e.g., when plugging them into Eq. 4.11 and Eq. 4.10. This alternate method renders tuning the β hyperparameter unnecessary. We term this variant "LogVar", and validate this approach through experimentation.

This experiment compares the MRL model trained using the *softplus* activation with the LogVar variant. We stress that we do not expect improved performance; this is a variant that functions similarly to the original model. The results, reported in Table

Table 4.6: Retrieval results for MRL when trained with 512 batch size and for different numbers of training steps. Training time is measured in hours. The retrieval performance is measured with MRR@10 for MS MARCO and NDCG@10 for the rest of the datasets.

Steps	Training time (h)	MS MARCO	TREC-DL'19	TREC-DL'20	SciFact	FiQA	TREC-COVID	CQADupStack
5880	79	.312	.680	.641	.365	.226	.407	.278
11760	162	.330	.693	.656	.517	.249	.457	.293

Table 4.7: MRL Variant: LogVar vs Softplus: LogVar performs similarly to Softplus but requires no hyperparameter tuning. Based on a two-tailed paired t-test ($p < 0.05$), there are no statistically significant differences between the models results. The retrieval performance is measured with MRR@10 for MS MARCO and NDCG@10 for the rest of the datasets.

Variant	MS MARCO	TREC-DL'19	TREC-DL'20	SciFact	FiQA	TREC-COVID	CQADupStack
Softplus	.373	.707	.681	.591	.291	.497	.327
LogVar	.373	.709	.679	.590	.296	.471	.327

4.7, show that the LogVar model achieves similar performance across datasets and metrics. We also conducted an ablation with different initializations as well as without distillation and observed the same trend. In essence, the LogVar model provides similar results to the Softplus model without requiring an additional hyperparameter.

4.5.6 Limitations of our study

Lack of details As mentioned previously, our reproducibility effort was hampered by a lack of crucial details in the original paper, such as the composition of the batches used while training the model, or which underlying cross-encoder was used as a teacher. Therefore, it is unclear whether the impressive performance improvements reported in the original paper are the result of some crucial implementation detail that was omitted. However, we tried to mitigate this limitation through *exhaustive* experimentation. The most promising part of that experimentation is presented in our ablation study, but it still constitutes a fraction of all model configurations that we attempted in order to get closer to the reported performance.

Batch size As mentioned in Section 4.5.4, we rely on GradCache [45] to support training MRL with larger batches. As a result, training time when training with a larger batch size does not decrease training time since we are still limited to a single

GPU. In contrast, training time increases significantly due to GradCache's overhead of calculating and storing gradients of representations. Due to this limitation, we cannot fully explore training MRL with a batch of 512 for longer (i.e., more training steps).

4.6 Conclusion

In this chapter, we reproduce the multivariate representation learning framework by Zamani and Bendersky [187] in a memory-constrained environment. After addressing a likely typographical error in the original paper's derivations, we show that in a fair comparison, MRL achieves similar performance to baseline models. Crucially, the claim that MRL significantly outperforms baseline models in out-of-domain datasets does not hold in our experimental setup. While MRL does not outperform baselines, we maintain that it remains a competitive retrieval model. We also conduct an extensive analysis of the predicted variance. Against our expectations, our analysis reveals that the variance vectors do not consistently express uncertainty. To add to the results of the original study, we conduct a thorough ablation study, investigating the impact of the different components of the MRL framework: (i) multivariate representations, (ii) distillation, and (iii) model initialization. Through this study, we conclude that multivariate representations do not improve or harm performance significantly, and knowledge distillation is the primary source of improvement. In addition, we show that models trained with an increased batch size using gradient caching methods and without increasing the number of training epochs are impacted negatively by the presence of in-batch negatives.

While we are unable to reproduce the results from the original paper, we maintain that the ideas in the original paper are very valuable to the community. Prior to this paper, uncertainty was only used in ranking, not first-stage retrieval. Representing KL divergence as a dot product allows for the incorporation of uncertainty in first-stage retrieval. This implies that *any* model that produces a distribution for queries/passages can be used in this framework, even if it was not trained with the objective function outlined in the paper – this is a promising direction for future research. Future work could further consider incorporating *document* uncertainty to the framework, e.g., for post-retrieval QPP.

This chapter concludes the first part of the thesis, which focuses on improving the robustness of neural retrievers against typos. Next, we address a different research theme, namely improving the robustness of speech-based search with neural retrieval.

Appendix 4.A

4.A.1: Dot product formulation in the original paper

For completeness, we include the original dot product formulation of the KL divergence below. Consider the term $\mathrm{tr}\{\Sigma_D^{-1}\,\Sigma_Q\}$, the source of the error. Zamani and Bendersky

[187] formulate this as:

$$\mathrm{tr}\{\Sigma_D^{-1}\,\Sigma_Q\} = \frac{\prod_{i=1}^{k}\sigma_{i_Q}^2}{\prod_{i=1}^{k}\sigma_{i_D}^2}. \tag{4.13}$$

This differs from our version in Eq. 4.6. This error is propagated throughout the next steps. The KL formulation becomes:

$$\begin{aligned}\mathrm{KLD}(Q \mid D) = \sum_{i=1}^{k}\log\sigma_{i_D}^2 + \frac{\prod_{i=1}^{k}\sigma_{i_Q}^2}{\prod_{i=1}^{k}\sigma_{i_D}^2} + \sum_{i=1}^{k}\frac{\mu_{i_Q}^2}{\sigma_{i_D}^2} \\ - \sum_{i=1}^{k}\frac{2\mu_{i_Q}\mu_{i_D}}{\sigma_{i_D}^2} + \sum_{i=1}^{k}\frac{\mu_{i_D}^2}{\sigma_{i_D}^2}.\end{aligned} \tag{4.14}$$

The equivalent dot product formulation, after taking the negative of the equation above gives us the original formulation (Eq. (14) in Zamani and Bendersky [187], with signs flipped for $\vec{d}$):

$$\vec{q} = \Big[1, \prod_{i=1}^{k}\sigma_{i_Q}^2, \mu_{1_Q}^2, \ldots, \mu_{k_Q}^2, \mu_{1_Q}, \ldots, \mu_{k_Q}\Big], \tag{4.15}$$

$$\vec{d} = \left[\gamma_D, \frac{1}{\prod_{i=1}^{k}\sigma_{i_D}^2}, \frac{1}{\sigma_{1_D}^2}, \ldots, \frac{1}{\sigma_{k_D}^2}, \frac{2\mu_{1_D}}{\sigma_{1_D}^2}, \ldots, \frac{2\mu_{k_D}}{\sigma_{k_D}^2}\right], \tag{4.16}$$

where γ_D is equivalent in our formulation i.e., Eq. 4.9. Here, $\vec{q}, \vec{d} \in \mathbb{R}^{1\times(2k+1)}$ and $q^\intercal \cdot d$ is equal to Eq. 4.14. Note that the final vector for $\vec{d}$, the signs are flipped because the similarity function is the *negative* KL divergence.

4.A.2: Dot product formulation of full KL divergence

We start from the *unsimplified* version of Eq. 4.8, that includes constants:

$$\begin{aligned}\mathrm{KLD}(Q \mid D) = \frac{1}{2}\Bigg[\sum_{i=1}^{k}\log\sigma_{i_D}^2 - \sum_{i=1}^{k}\log\sigma_{i_Q}^2 - k + \sum_{i=1}^{k}\frac{\sigma_{i_Q}^2}{\sigma_{i_D}^2} \\ + \sum_{i=1}^{k}\frac{\mu_{i_Q}^2}{\sigma_{i_D}^2} - \sum_{i=1}^{k}\frac{2\mu_{i_Q}\mu_{i_D}}{\sigma_{i_D}^2} + \sum_{i=1}^{k}\frac{\mu_{i_D}^2}{\sigma_{i_D}^2}\Bigg].\end{aligned} \tag{4.17}$$

To formulate it as a dot product, we use the definition of the document prior, γ_D, from Eq. 4.9, but further define γ_Q as:

$$\gamma_Q = \sum_{i=1}^{k}\log\sigma_{i_Q}^2. \tag{4.18}$$

Then we can extend the vector representations in Eq. 4.10 and Eq. 4.11 to include γ_Q:

$$\vec{q'} = \Big[1, \gamma_Q, \sigma^2_{1_Q}, \ldots, \sigma^2_{k_Q}, \mu^2_{1_Q}, \ldots, \mu^2_{k_Q}, \mu_{1_Q}, \ldots, \mu_{k_Q}\Big], \tag{4.19}$$

$$\vec{d'} = \left[\gamma_D, 1, \frac{1}{\sigma^2_{1_D}}, \ldots, \frac{1}{\sigma^2_{k_D}}, \frac{1}{\sigma^2_{1_D}}, \ldots, \frac{1}{\sigma^2_{k_D}}, -\frac{2\mu_{1_D}}{\sigma^2_{1_D}}, \ldots, -\frac{2\mu_{k_D}}{\sigma^2_{k_D}}\right], \tag{4.20}$$

where $\vec{q'}, \vec{d'} \in \mathbb{R}^{1\times(3k+2)}$. Then,

$$\mathrm{KLD}(Q \mid D) = \frac{1}{2}\big(\vec{q'}^{\intercal} \cdot \vec{d'} - k\big), \tag{4.21}$$

should precisely yield the KL divergence between the distributions of Q and D, as defined in Eq. 4.4.

4.A.3: Training setup for CLDRD

One of the perks of training with knowledge distillation is that it can leverage the incomplete relevance judgments that most large-scale retrieval datasets suffer from. Hence, it can rely only on the teacher supervision signal, i.e., training data is generated and scored by the teacher model without human assessments. Furthermore, combining knowledge distillation with in-batch negative training can be impractical when the teacher is a computationally expensive cross-encoder [91]. On the other end of the spectrum, even though the exact relevance scores produced by the teacher model do not impact the loss value (since the loss only considers the order of passages) they still play an important role in controlling which query-passage pairs will contribute to the loss (term $\mathbb{1}\{y^t_q(d) > y^t_q(d')\}$ in the loss). When the raw scores from the teacher model are used, all pairs contribute to the loss, even when contrasting two irrelevant passages.

In contrast to the training setup used by Zamani and Bendersky [187] (see Section 4.3.2), the work by Zeng et al. [188]–the work that initially proposed the listwise distillation loss used by Zamani and Bendersky [187] and the primary competing approach of MRL–uses pseudo-labeling to create the batch and does not rely on in-batch negatives. The authors use the following approach to compute the listwise distillation loss:

- Given a query q, the passage set D_q is constructed with respect to the top-k passages in the ranked list returned by the teacher model (reranking order). In particular, the first K passages in the ranked list returned by the teacher model are considered positive, the next K' are considered hard negatives, and the remaining K'' soft negatives.
- $y^t_q(d)$ is a relevance label according to the group, passage d belongs to:

$$y^t_q(d) = \begin{cases} \frac{1}{r^t_{qd}} & \text{iff d is positive} \\ 0 & \text{iff d is hard-negative} \\ -1 & \text{iff d is soft-negative} \end{cases}$$

  where r^t_{qd} is the ranking position of the document d given the query q in the teacher ranked list.

Table 4.8: Statistics and Description of Evaluation Datasets. The average number of tokens for queries ("q") and passages ("p") was computed based on the `distilbert-base-uncased` tokenizer.

	Name	Domain	# q	# p	avg. q length	avg. p length
ID	MS MARCO Dev	Miscellaneous	6,890	8,841,823	9.01	76.97
	TREC-DL 19	Miscellaneous	43	8,841,823	9.02	76.97
	TREC-DL 20	Miscellaneous	54	8,841,823	9.22	76.97
OOD	Scifact	Scientific Document Retrieval	300	5,183	22.84	315.65
	FiQA	Financial QA	648	57,638	15.59	177.11
	TREC-COVID	Biomedical document retrieval	50	171,332	18.04	224.78
	CQADupStack	Community QA retrieval	13,145	457,199	13.55	248.73

Table 4.9: Hyperparameter search space for the models we experiment with. Negatives are presented in the following format:$[\#ANN\ negatives, \#BM25\ negatives]$

Parameter	Values
Learning rate	1×10^{-4}, 1×10^{-5}, 1×10^{-6}, 3×10^{-6}, 5×10^{-6}, 7×10^{-6}
β	0.5, 1, 2.5, 7.5
Negatives	[5, 25], [10, 20], [15, 15], [20, 10], [25, 5]

4.A.4: Datasets

In Table 4.8 we present the statistics of the datasets.

4.A.5: Hyperparameters

In Table 4.9 we detail our hyperparameter search space used throughout our experimental setup. For MRL we search w.r.t. learning rate, β, and negatives. The best parameters were: $\beta = 2.5$, $lr = 5 \times 10^{-6}$, 5 negative passages from BM25, and 25 negative passages from the student model. For MRL-CLDRD we search w.r.t. learning rate and β. The best parameters were: $\beta = 2.5$, and $lr = [5 \times 10^{-6}, 1 \times 10^{-6}, 1 \times 10^{-6}]$ for the three curriculum iterations. For CLDRD the best learning rates for each of the three curriculum iterations were $[7 \times 10^{-6}, 3 \times 10^{-6}, 3 \times 10^{-6}]$, while for DPR the best learning rate was 7×10^{-6}.

Our academic computational budget allowed us to perform a hyperparameter search only on the models in Tables 4.1 and 4.2 as well as the model using cross-entropy without distillation in Table 4.5. However, we note that the model performance is quite robust to the choice of hyperparameters in our initial experiments – mitigating this limitation to an extent.

Part II

Improving the Robustness of Speech-based Search with Neural Retrieval

5

Dealing with Speech Recognition Errors for Dense Retrieval

In the first part of this thesis, we focused on textual queries. However, in today's world, millions of users use speech interfaces to interact with search engines. Voice interaction has become very popular because it is convenient, hands-free, provides a natural user experience and supports users with visual/motor impairments who cannot use conventional text entry mechanisms like a keyboard. Thus, millions of users are voicing their queries instead of typing them.

In this part of the thesis, we focus on spoken queries. A straightforward retrieval approach to tackle spoken queries combines an automatic speech recognition (ASR) system with a text retriever. In such a pipeline approach, the spoken queries are transcribed with an ASR model and then passed to the text retriever. However, the ASR system may not always provide a perfect transcription. Difficult accents, background noise, and rare entities can all contribute to a corrupted transcription. In this chapter, we aim to answer **RQ4**: Are dense retrievers robust against queries that contain transcription errors?

5.1 Introduction

Nowadays users interact with a wide range of commercial question answering (QA) systems via speech interfaces. Millions of users are voicing their questions on virtual voice assistants such as Amazon Alexa, Apple Siri, or Google Assistant through their smart devices. Such voice assistants do not only increase the convenience with which users can query them but can support users with visual and motor impairments for which the use of conventional text entry mechanisms (keyboard) is not applicable [121]. Despite the popularity of voice assistants among users globally and the advancements in spoken-language understanding [7, 40], there are surprisingly limited efforts in studying spoken QA and its limitations.

The majority of research focuses on reading comprehension as a component of spoken QA [38, 85, 126]. Previous works studied the case where the provided question

This chapter was published as G. Sidiropoulos, S. Vakulenko, and E. Kanoulas. On the impact of speech recognition errors in passage retrieval for spoken question answering. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, 2022.

includes noise introduced by an automated speech recognition (ASR) system; audio is converted to text before reading comprehension is performed. Ravichander et al. [126] showed that ASR noise not only dramatically affects the performance of transformer-based reading comprehension models but also that it is a more challenging type of noise than the noise generated from keyboard mistyping or faulty machine translation. Faisal et al. [38] showed that background differences in users, such as their accents, can affect the performance of reading comprehension models differently.

Even though robustifying reading comprehension against ASR noise is essential for extracting the answer to a question, as a subsequent step of passage retrieval, it is bounded by the ability of the QA system to retrieve the relevant passages. A typical QA pipeline consists of an efficient retriever that reduces the search space from millions of passages to the top-k and a reader that extracts the answer. Dense passage retrieval has become a new paradigm to retrieve relevant passages, setting the state-of-the-art performance in several leaderboards [127, 154]. Inferior retrieval of the relevant passages will negatively affect the performance of the overall system.

Typically, state-of-the-art dense retrieval models are evaluated on clean datasets with noise-free questions [73, 127, 154]. However, questions posed to real-world QA systems are prone to errors. Therefore, these models will encounter noisy questions when deployed in real-world applications, affecting their performance. User-generated textual questions can include typos such as keyboard typos due to fast typing, misspellings, and phonetic typing errors (for words with close pronunciation). Recent works showed that even state-of-the-art dense retrieval models are not robust against simple typos [139, 196, 197]. Sidiropoulos and Kanoulas [139] showcased the dense retrievers' lack of robustness to typos in the question and proposed a combination of data augmentation with a contrastive loss to robustify the model; see Chapters 2 and 3. Zhuang and Zuccon [197] increased the robustness of dense retrievers against typos by replacing the WordPiece tokenizer, which is extremely sensitive to typos, with the Character-CNN module and further combined it with a knowledge distillation method.

On the other end of the spectrum, spoken questions voiced by users are also vulnerable to errors due to the ASR systems that convert them to text. How the presence of ASR noise in questions affects retrieval models has not been studied yet. In this chapter, we address the need for evaluating passage retrieval for spoken QA. To the best of our knowledge, this is the first work in this direction.

Since there is no available dataset for passage retrieval where questions have ASR noise, we simulate ASR noise by automatically transcribing synthetically voiced questions. We then compare the robustness of lexical and dense retrievers by evaluating them against questions with and without ASR noise. Preliminary results showed that neither lexical nor dense models are effective against questions with ASR noise leading to a significant drop in retrieval performance. We find that using data augmentation with ASR noise to train a dense retriever is a promising approach for increasing robustness against ASR noise. However, the generation of such synthetic data is time-consuming and limited to the languages/accents supported by the text-to-speech system. We explore if typo augmentation (faster and not bound to specific accents/languages) can alleviate these limitations. Our experimental results show that typo robust dense retrievers can increase robustness against ASR noise to some extent; however, ASR data augmentation remains a significantly more effective approach. Since users can have different local

accents, we experiment with multiple accents of the same language and unveil that users' accents play an important role in retrieval performance. Finally, to study a real-world scenario with voice variation and non-native speakers voicing questions, we also build a new test set where the questions have natural ASR noise. This set consists of 700 questions voiced by human annotators.

To tackle **RQ4** we aim to answer the following research sub-questions:

RQ4.1 What is the impact on the performance of lexical and dense retrievers when questions have ASR errors?

RQ4.2 Are typo-robust dense retrieval approaches also robust against ASR noise? How competitive are they against dense retrieval trained via data augmentation with ASR noise?

RQ4.3 Do certain accents affect the effectiveness of the retriever more than others?

RQ4.4 Does natural ASR noise affect the robustness of dense retrievers more than synthetic ASR noise?

Our main contributions can be summarized as follows: (i) we provide two large-scale datasets where questions have synthetic ASR noise to facilitate research (evaluation and training of new models) on passage retrieval for spoken QA, (ii) we create a new, challenging test set that contains 700 questions with natural ASR noise, (iii) we show how lexical and dense retrievers are not robust against ASR noise and propose data augmentation for robustifying the latter, and (iv) we study how performance varies with respect to different accents and synthetic vs. natural spoken questions.[1]

5.2 Experimental Setup

5.2.1 Datasets and evaluation metrics

For our experiments, we focus on two large-scale datasets, namely, MS MARCO passage ranking [113] and Natural Questions (NQ) [78]. In MS MARCO the objective is to rank passages based on their relevance to a question. The questions were compiled from Bing search logs, while the underlying corpus consists of 8.8 million passages. NQ is an open-domain QA dataset where questions were sampled from Google search logs and can be answered over Wikipedia.

For MS MARCO, to measure the retrieval performance, we use the official metric MRR@10 alongside Recall (R). We report the metrics on the development set, MS-MARCO (Dev), since the ground-truths for the test set are not available to the public. Similar to previous works on NQ, we report answer recall (AR) at the top-k retrieved passages. Answer recall evaluates whether the ground-truth answer string appears among the top retrieved passages.

[1] https://github.com/GSidiropoulos/passage_retrieval_for_spoken_qa

Table 5.1: WER for transcribed synthetic and natural spoken questions. For synthetic, we report on the whole NQ (test) while for the natural on a 700-question subset of NQ (test).

	Synthetic				
	en-US	en-AU	en-IN		
NQ	20.70	21.01	24.77		
	Natural				
	A_1	A_2	A_3	A_4	avg
NQ	38.82	38.16	44.01	35.28	39.60

5.2.2 Simulating ASR noise

To study the impact of speech recognition errors in passage retrieval for spoken QA, we need a large dataset of questions with ASR noise. There is no such dataset publicly available and hence in this work we build one. Following previous works [38, 85, 126], to simulate ASR noise, we follow a pipeline that consists of speech generation by a text-to-speech system and transcription of the generated speech by a speech-to-text system. We obtain the spoken version of the original questions via Google TTS and their transcriptions using *wav2vec 2.0* [5]. We use English as the system language. In particular, for U.S. English (en-US), the word error rate (WER) is 20.70 and 34.26 for NQ and MS MARCO, respectively. However, as users can have different local accents, we experiment with other English variations supported by Google TTS, such as Australian English (en-AU) and Indian English (en-IN). We report their word error rate scores in Table 5.1. Common errors in the transcribed questions include incorrect splits and phonetical spelling of relatively rare words such as entity mentions.

5.2.3 Natural ASR noise

To simulate a natural real-world setting, we manually construct a dataset with natural ASR noise. In order to create a dataset with natural ASR noise we use the SANTLR [89] speech annotation toolkit. Specifically, we use SANTLR to record spoken versions of the question from four human annotators. Subsequently, we transcribe the obtained recordings using *wav2vec 2.0* (similar to Section 5.2.2). We obtain audio recordings (in English) for spoken versions of 700 questions from the NQ dataset, voiced by four human annotators. The annotators are instructed to (i) read the prompt question, (ii) ensure that they can pronounce every word appearing in the question or move to the next one, and (iii) finally record. The annotators consist of a French female (A_1), a Greek male (A_2), an Indian male (A_3), and a Russian female (A_4), with the first three voicing 200 unique questions each and the last voicing 100 unique questions. All annotators are using English in their everyday life. We use a mixture of accents originating from non-native English speakers to resemble a real-world scenario. Voice assistants do not support the majority of the world's languages [40]. Therefore, many users have to voice their questions in a language different from their native one. WER scores can be found in Table 5.1. Similar to synthetic noise, the most common errors include incorrect splits

and the phonetical spelling of entity mentions. However, these errors are significantly more prominent in the case of natural noise.

5.2.4 Models

BM25 BM25 is a standard retrieval model based on best match; there is lexical overlap between the query and every retrieved passage. We use the Anserini IR toolkit [177] to compute BM25 scores.

DR A typical dense retriever [73] is a dual-encoder BERT-based model used for scoring question-passage pairs. Given a question q, a positive (i.e., relevant) passage p^+ and a set of negative (i.e., irrelevant) passages $\{p_1^-, p_2^-, \ldots, p_n^-\}$, the model learns to rank the positive question-passage pair higher than the negative ones. The two separate encoders of the model are fine-tuned via the minimization of the softmax cross-entropy:

$$\mathcal{L}_{CE} = -\log \frac{e^{s(q,p^+)}}{e^{s(q,p^+)} + \sum_{p^-} e^{s(q,p^-)}}. \tag{5.1}$$

During inference time, the similarity of a question-passage pair is calculated as the inner product of the respective question embedding and passage embedding. The whole corpus is encoded into an index of passage vectors offline, and retrieval with respect to a question is implemented with efficient maximum inner product search [70] over the index. We follow the dual-encoder architecture rather than a cross-encoder one (that jointly encodes question and passage) due to its high efficiency as a first-stage ranker in large-scale settings. While the latter can achieve higher performance, the former makes the whole corpus indexable.

DR+Data augm. A dense retriever with data augmentation that builds on the standard practice for improving the robustness of neural models by augmenting the training data with noisy data. We explore two cases of data augmentation, namely, augmentation with synthetic keyboard noise and augmentation with synthetic ASR noise. For the former, we augment each question on the training set with keyboard noise following the approach presented in [139], while for the latter, we augment with ASR noise as shown in Section 5.2.2. In contrast with ASR noise, keyboard noise will rarely alter the original word into a different correctly spelled word. For example, the question "who is the owner of reading football club" can be transformed to "who is the owner of retting football club" if augmented with ASR noise and to "who is the ownrr of reading football club" if augmented with keyboard noise; "retting" is a correctly spelled word, while "ownrr" is not.

DR+CharacterBERT+ST The dense retriever with CharacterBERT and self-teaching [197] is the current state-of-the-art dense retrieval approach for dealing with typos. It builds on DR by altering the backbone BERT encoder with CharacterBERT and further uses an effective training method that distills knowledge from questions without typos into the questions with typos, known as self-teaching. Specifically, the goal of the latter is to minimize the difference between the similarity score distribution from the question

Table 5.2: Retrieval results for the settings of (i) clean questions (Original) and (ii) questions with synthetic ASR noise (ASR); synthetic voice with a U.S. English accent. Statistical significance difference determined with a paired t-test ($p < 0.05$) BM25=b, DR=d; DR with typo data augm.=t; DR+CharBERT+ST=c. Note that we use the pretrained DR+CharBERT+ST from the original paper.

	Noise	NQ (Test)					
		Original			ASR		
		AR@5	AR@20	AR@100	AR@5	AR@20	AR@100
BM25	-	40.94	57.81	70.83	23.32	36.98	52.49
DR	-	66.26	77.75	85.26	41.91	54.65	67.03
DR+Data augm.	Typos	67.47	78.75	85.40	46.75	60.72	71.55
DR+CharBERT+ST [197]	Typos	-	-	-	-	-	-
DR+Data augm.	ASR	66.67	78.00	85.45	**54.84**^{bdtc}	**67.89**^{bdtc}	**78.50**^{bdtc}
	Noise	MS MARCO (Dev)					
		Original			ASR		
		R@50	R@1000	MRR@10	R@50	R@1000	MRR@10
BM25	-	59.11	85.61	18.67	24.71	45.34	6.97
DR	-	74.58	94.19	28.69	35.31	56.83	12.13
DR+Data augm.	Typos	75.17	94.54	29.10	46.75	64.57	13.02
DR+CharBERT+ST [197]	Typos	77.55	94.95	32.51	45.45	68.20	16.35
DR+Data augm.	ASR	73.47	93.96	29.14	**54.48**^{bdtc}	**81.25**^{bdtc}	**18.43**^{bdtc}

with the typo and the score distribution from the question without the typo. This is achieved by minimizing the KL-divergence:

$$\mathcal{L}_{KL} = \tilde{s}(q', p) \cdot \log \frac{\tilde{s}(q', p)}{\tilde{s}(q, p)}, \tag{5.2}$$

where q' represents the typoed question and $\tilde{s}$ the softmax normalized similarity score. The final loss is the sum of the $\mathcal{L}_{KL}$ (5.2) and $\mathcal{L}_{CE}$ (5.1) losses.

5.2.5 Implementation details

The DR model we use in our experiments is trained using the in-batch negative setting described in [73]. The question and passage BERT encoders are trained using Adam with a learning rate of $2e$-5 and linear scheduling with warm-up rate of 0.1 for (i) 40 epochs with a batch size of 64 for the case of NQ and (ii) 10 epochs with a batch size of 84 for MS MARCO. Moreover, we use the same hyper-parameters when training DR with data augmentation. For DR+characterBERT+ST, we use the pre-trained model as provided by the authors of [197]. The audio input is sampled at 16Khz to be compatible with the transcription model we use (*wav2vec*).

5.3 Results

In this section, we present our experimental results to answer our research questions. To answer **RQ4.1**, we compare the retrieval performance of a lexical retriever (BM25) and

Table 5.3: Retrieval results for DR trained with ASR data augmentation (questions augmented with U.S. English synthetic ASR noise).

# Additional training questions with ASR noise	AR@5	AR@20	AR@100
40K	54.84	67.89	78.50
4K	48.69	61.96	73.35
400	44.29	57.28	69.27
0	41.91	54.65	67.03

a dense retriever (DR) for the settings of clean questions and questions with synthetic ASR noise. As we can see from the first two rows in Table 5.2, DR significantly outperforms BM25 across both settings (original and ASR). On the other hand, when questions have ASR noise, there is a dramatic drop in performance for both BM25 (MRR@10 drops 62.66% and AR@5 43.03%) and DR (MRR@10 drops 57.72% and AR@5 36.74%). This drop indicates the lack of robustness of lexical and dense models against ASR noise.

Data augmentation with typos and self-teaching for knowledge distillation from questions without typos into the questions with typos are two training schemes for robustifying DR. For **RQ4.2**, we examine how these two perform compared to DR with standard training on clean questions on the ASR noise scenario. For rows using "Typos" as noise in Table 5.2, we notice that the models can increase robustness against ASR noise, to some extent, even though the typos are not originating from the same distribution as the ASR noise during inference. That said, as was expected, DR holds the best results when augmenting the training set with ASR noise.

Furthermore, we investigate how the retrieval performance varies depending on the number of available questions with ASR noise used to augment the training set. Table 5.3 shows the results. We observe a significant increase in performance even in the low data regime, with 400 noisy questions. Intuitively, there is consistent improvement in performance as the additional data increase.

At this point, we have seen that DR with ASR data augmentation is an effective approach for dealing with spoken questions. For **RQ4.3**, we want to study the impact of different English accents during inference. To do so, we use the DR model which is augmented with synthetic ASR noise from U.S. English and test it against different spoken accents such as Australian English (en-AU) and Indian English (en-IN). The results in Table 5.4 show that different accents have different impacts on performance. Specifically, we observe a small drop in performance for Australian English while the drop is more prominent for Indian English. This is strongly related to the fact that U.S. English is phonetically more similar to Australian English than to Indian English [43, 92, 156].

We underline that using synthetic ASR noise in the respective English variation for augmenting the training set could help boost performance. However, this is not a viable solution if we take into account that (i) such an approach would require training a new system for every new variation and to the extreme for each individual, and (ii) there is a

Table 5.4: Retrieval results for DR trained with ASR data augmentation (augmented with U.S. English synthetic ASR noise) and tested against synthetic ASR for various accents.

Data augm.	Test	AR@5	AR@20	AR@100
en-US	en-US	54.84	67.89	78.50
en-US	en-AU	53.54	66.70	76.92
en-US	en-IN	46.37	60.27	71.85

Table 5.5: Retrieval results for DR against questions with ASR noise from (i) synthetic voice with a U.S. English accent (Synthetic), and (ii) natural voice with a mixture of French, Greek, Indian and Russian accents (Natural). We use the same subset of 700 questions to ensure a fair comparison.

Data augm.	ASR Synthetic			ASR Natural		
	AR@5	AR@20	AR@100	AR@5	AR@20	AR@100
-	38.88	53.65	65.85	16.64	28.55	42.46
Typos	46.91	59.82	71.87	24.82	35.86	50.50
ASR	55.38	67.43	77.76	29.98	42.32	55.66

limitation in the available synthetic voice accents.

In a real-world scenario, alongside voice variation, accents can greatly vary depending on the origin of the user; since non-native English speakers are voicing their questions in English when interacting with voice assistants. To showcase the real-world utility of dense retrieval (**RQ4.4**), we evaluate DR and its data augmented variations on our natural ASR noise dataset (see Section 5.2.3). As we can see in Table 5.5, even though both synthetic and natural ASR noise decrease the retrieval performance, natural ASR noise appears to be a significantly more challenging setting. Despite the fact that the distribution of synthetically generated ASR noise and keyboard noise differs from that of natural ASR, we notice that DR combined with data augmentation achieves better results than DR alone. We find that DR with synthetic ASR data augmentation outperforms its typo counterpart. Unfortunately, synthetic ASR and typo noise are inefficient while they do not generalize well to natural ASR noise.

5.4 Conclusions

In this chapter, we study the impact of speech recognition errors in passage retrieval for spoken QA, following an ASR-Retriever pipeline approach (i.e., the spoken question is passed through an ASR system and its transcription is fed to a text retriever). We showcase that the effectiveness of lexical and dense retrievers drops dramatically when dealing with transcribed spoken questions. Moreover, we explore how typo-robust dense retrieval approaches perform against questions with ASR noise. Even though they can increase robustness compared to standard training on clean questions only, dense

retrieval trained via data augmentation with ASR noise is a more effective approach. Finally, we compare the effect of synthetic vs. natural ASR noise and find that the latter is a significantly more challenging setting. Unfortunately, data augmentation with synthetic ASR noise does not generalize well to the natural ASR scenario.

One important shortcoming of the ASR-Retriever pipeline is that the ASR model will propagate its errors on the retriever. Hence, the retrieval performance is limited by the quality of the transcriptions. Furthermore, ASR-Retriever pipelines require that there is enough annotated speech data to train an effective ASR model. For future work, we plan to build on our insights and develop more sophisticated approaches for robustifying dense retrievers against ASR noise. Also, we aim to investigate ways to produce noise that can closely resemble natural ASR noise.

In this chapter, we studied speech-based search from an ASR-Retriever pipeline point of view. In the following chapter, our focus shifts to eliminating the need for ASR and enabling end-to-end training.

6

Multimodal Dense Retrieval for Spoken Queries

In the previous chapter, we adopted a pipeline approach consisting of an automatic speech recognition (ASR) model that transcribes the spoken question before feeding it to a dense text retriever. Such *ASR and Retriever* pipelines have several limitations. The need for an ASR model limits the applicability to low-resource languages and specialized domains with no annotated speech data. Furthermore, the ASR model propagates its errors to the retriever. In contrast, an ASR-free, end-to-end trained multimodal retriever that can work directly on spoken queries might be able to alleviate some of the aforementioned shortcomings. In this chapter, we aim at answering **RQ5**: How does a multimodal dense retriever perform in speech-based search?

6.1 Introduction

Voice assistants are convenient, easy to use, and can support users with visual and motor impairments that cannot use conventional text entry devices. Voice assistants such as Amazon Alexa, Apple Siri, and Google Assistant are used daily by everyday users. As a result, nowadays, users interact with a wide range of commercial Question Answering (QA) systems via such speech interfaces. In other words, millions of users are voicing their questions on virtual voice assistants instead of typing them. That has led to the emergence of speech-based open-domain QA, a QA task on open-domain textual datasets where questions are in speech form.

Most of the research in speech-based open-domain QA focuses on reading comprehension as a component of the commonly adopted *retriever and reader* framework [38, 126]. However, for effective answer extraction, we need an effective retriever that reduces the search space from millions of passages to the $top\text{-}k$ most relevant; the performance of passage retrieval bounds that of reading comprehension. Hence, studying passage retrieval in speech-based open-domain QA is crucial. Despite the attention that passage retrieval for traditional open-domain QA receives from the community [194], there are surprisingly few efforts in studying passage retrieval for speech-based

This chapter was published as G. Sidiropoulos and E. Kanoulas. Multimodal dense passage retrieval for open-domain spoken question answering. *Under submission*.

open-domain QA.

Recently, Sidiropoulos et al. [144] studied passage retrieval for speech-based open-domain QA. In particular, following a pipeline approach consisting of an automatic speech recognition (ASR) model for transcribing the spoken question and a dense retriever[1] for text retrieval (*ASR-Retriever*), they investigated the effectiveness of dense retrievers on questions with ASR mistranscriptions. In their work, they build on the assumption that the clean textual version of the spoken question is available at training, and the ASR mistranscriptions appear only at inference. That said, such an approach is not directly applicable in a real-world scenario where only the spoken questions are available during training. In a real-world scenario, following an *ASR-Retriever* pipeline would require either (i) using the ASR model to produce the training questions or (ii) training on a different dataset where questions in text form are available.

The *ASR-Retriever* pipeline has several limitations. First and foremost, it does not support training in an end-to-end manner. As a direct consequence, ASR propagates its errors to the downstream retriever; the higher the word error rate from ASR, the higher the negative impact on the performance of the retriever. ASR models suffer from mistranscribing long-tail named entities and domain-specific words [68, 99, 165]. The former is of great importance when working on questions because corrupting the main entity of a question can dramatically affect retrieval [144]. For example, when the question "what is the meaning of the name sinead?" is transcribed as "what is the meaning of the name chinade?", then the retrieval will be centered around the wrong entity "chinade". Additionally, training an ASR model requires obtaining a large amount of annotated speech which is not always available (think for example of low-resource languages). Finally, during query time, feeding the spoken question to the transformer-based ASR model to obtain its transcription before performing retrieval results in high query latency.

In contrast to the *ASR-Retriever* pipeline, we propose a multimodal dual-encoder dense retriever that does not require an ASR model and can be trained end-to-end. Our method adapts the dual-encoder architecture used for dense retrieval in traditional open-domain QA (i.e., where both question and passage are in text form) [73] by replacing the backbone language model used to encode the questions with a self-supervised speech model (Figure 6.1). To the best of our knowledge, this is the first multimodal dual-encoder approach for speech-based open-domain QA. Furthermore, in this chapter, we benchmark passage retrieval for the speech-based open-domain QA using *ASR-Retriever* pipeline approaches, where the spoken questions are transcribed with an ASR model and then used for training the dense text retrievers.

To answer **RQ5** we break it down into four research suq-questions:

RQ5.1 How does our ASR-free, end-to-end trained multimodal dense retriever perform compared to its *ASR-Retriever* pipeline counterparts?

RQ5.2 Can our ASR-free retriever alleviate the limitations of *ASR-Retriever* pipelines when it comes to ASR mistranscribing important words in a question?

[1] Dense text retrievers use transformers to encode the question and passage into a single vector each and further use the similarity of these vectors to indicate the relevance between the question and passage.

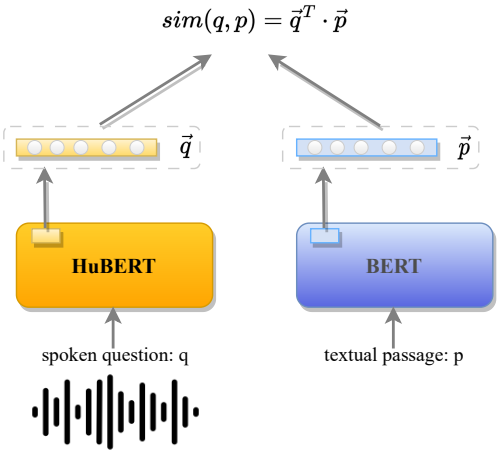


Figure 6.1: Overview of our multimodal dense retriever.

RQ5.3 How does our ASR-free retriever perform against *ASR-Retriever* when the latter has to deal with previously unseen ASR mistranscription?

RQ5.4 What is the most effective training scheme for learning a multimodal dense retriever?

We make the following contributions:

- We benchmark the speech-based open-domain QA task with stronger baselines. In detail, we train the retriever in the *ASR-Retriever* pipeline on spoken question transcriptions and not clean textual questions as in Sidiropoulos et al. [144].
- We propose an ASR-free, end-to-end trained multimodal dense retriever that can work on speech, and we demonstrate a setup for effective training.
- Experimental results show that our retriever is a promising alternative to *ASR-Retriever* pipelines and competitive on shorter questions.
- We demonstrate through thorough analysis the robustness of our approach compared to the *ASR-Retriever* pipeline, where the retrieval performance of the latter is strongly related to the ASR error; consequently, it varies significantly under different situations. We unveil that pipelines witness a dramatic drop in their retrieval performance (i) as the word error rate of the ASR model increases, (ii) when important words in the spoken question are mistranscribed, and (iii) on mistranscriptions that were not encountered during training. Our ASR-free retriever can alleviate these problems and yield better results.[2]

[2]We plan to release the code and pre-trained models for replicating all the experiments upon acceptance of the paper underlying this chapter.

6.2 Related Work

Passage retrieval is a key task in traditional open-domain QA and speech-based open-domain QA. Following the retriever and reader framework, the retriever reduces the search space for effective answer extraction and provides the support context that users can use to verify the answer. Traditional open-domain QA has seen significant advancements with the introduction of dense retrievers [73, 124, 194] that have demonstrated higher effectiveness than bag-of-words methods.

Despite their effectiveness, dense retrievers can still perform poorly in the presence of noise. Zhuang and Zuccon [196] investigated the robustness of dense retrievers against questions with typos. Their work suggested that dense retrievers perform poorly on questions that have typos, and to this end, they proposed a typo-aware training strategy (data augmentation) to alleviate this problem. Sidiropoulos and Kanoulas [139] built upon this and further proposed to combine data augmentation with a contrastive loss to bring closer the representations of a question and its typoed variants. Zhuang and Zuccon [197] showed that replacing the backbone BERT encoders of the dense retriever with CharacterBERT can increase robustness against typos. They further improved robustness via a novel self-teaching training strategy that distills knowledge from questions without typos into typoed questions.

Recently, Sidiropoulos et al. [144] studied the robustness of dense retrievers under ASR noise (see Chapter 5). By employing a pipeline approach with an ASR system and a dense retriever for text retrieval, they showed that dense retrievers, trained only on clean questions, are not robust against ASR noise. They further suggested that training the retriever to be robust against typos can increase the robustness against ASR noise. To the best of our knowledge, this is the first work that studies passage retrieval for speech-based open-domain QA.

On the other end of the spectrum, recent works in speech-based QA explored reading comprehension as a component of the retriever and reader framework. Ravichander et al. [126] showed that mistranscription from the ASR model leads to a huge performance degradation in the transformer-based reading comprehension models while Faisal et al. [38] suggested that background differences in the users (e.g., accents) have different impacts on the performance of reading comprehension models.

At this point, we want to highlight the differences between speech-based open-domain QA and spoken QA [85, 183] since there can be misconceptions due to the similarity in their names. Spoken QA is a searching through speech task where given a text question and a spoken document the goal is to find the answer from this single spoken document. Therefore, this is a fundamentally different problem from the problem we consider in this work.

6.3 Methodology

6.3.1 Problem definition

Let $p \in \mathcal{C}$ denote a passage in text form within a passage collection $\mathcal{C}$ in the scale of millions and q a question in speech form. In passage retrieval for speech-based

open-domain QA, given q and $\mathcal{C}$, the task is to retrieve a set of relevant passages $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, where $p_i \in \mathcal{C}$ and can answer q.

6.3.2 Multimodal dense retriever

In passage retrieval for speech-based open-domain QA, the *ASR-Retriever* pipelines that were used so far suffer from propagating ASR errors to the retriever. Such approaches are not trained end-to-end; thus, the quality of the ASR transcriptions bounds the retrieval performance. Furthermore, the requirement for an ASR model limits the applicability of such pipelines to scenarios where annotated speech is available for training the ASR model. To alleviate the above-mentioned problems, we propose an ASR-free multimodal dense retriever that can support speech, and can be trained end-to-end.

Specifically, we modify the dual-encoder architecture for dense text retrieval in traditional open-domain QA [73] to account for spoken questions. We replace the BERT-based question encoder with HuBERT [66] and leave the BERT-based passage encoder as is. HuBERT is a self-supervised model for speech representation learning, which uses a BERT-like masked prediction loss. It uses offline clustering to provide target labels for masked language model pre-training. Figure 6.1 illustrates the architecture of our ASR-free, multimodal dense retriever. Suppose a pair of question q, in speech form, and a passage p, in text form. The speech and language encoders produce the output representations:

$$HuBERT(q) = (\boldsymbol{q}_1, ..., \boldsymbol{q}_n), \;\; BERT(p) = (\boldsymbol{p}_1, ..., \boldsymbol{p}_m). \tag{6.1}$$

We use the first token embedding output from the speech and language modules to encode questions and passages into a single vector each:

$$\vec{q} = \boldsymbol{q}_1, \;\; \vec{p} = \boldsymbol{p}_1. \tag{6.2}$$

The relevance between a question and a passage is computed as the dot product of their vectors:

$$s(q, p) = \vec{q}^T \cdot \vec{p}. \tag{6.3}$$

We train our model so that relevant passages to the question (i.e., passages that include the answer) have a higher similarity score than the irrelevant passages. We followed the original dual-encoder training setting from Karpukhin et al. [73] where, given a question q, a relevant passage p^+ and a set of irrelevant passages $\{p_1^-, p_2^-, \ldots, p_n^-\}$, the model is fine-tuned via the minimization of the softmax cross-entropy:

$$\mathcal{L}_{CE} = -\log \frac{e^{s(q,p^+)}}{e^{s(q,p^+)} + \sum_{p^-} e^{s(q,p^-)}}. \tag{6.4}$$

The inference phase of our multimodal dense retriever remains the same as in traditional dual-encoders for dense text retrieval. Specifically, we compute the similarity of a question-passage pair as the inner product of the respective question embedding and passage embedding. At query time, only the question needs to be encoded. We build a dense index of passage vectors (offline) by encoding the whole corpus and storing

it in an index structure that supports efficient retrieval of the relevant passages via approximate nearest neighbor search [70]. At this point, we want to highlight that we choose a dual-encoder architecture because it has shown high efficiency as a first-stage ranker in large-scale settings. On the contrary, even though cross-encoder architectures can achieve higher performance due to jointly encoding questions and passages, they are not indexable and hence are re-rankers.

6.4 Experimental Setup

6.4.1 Baselines and implementation

We compare our proposed multimodal method against pipeline approaches that consist of (i) an ASR model for transcribing the spoken question and (ii) a retriever. In detail, similar to Sidiropoulos et al. [144] we use *wav2vec* 2.0 [5] pretrained and fine-tuned on 960 hours of annotated speech data [118] as the ASR model. Concerning the retrievers, we experiment with popular lexical and dense retrievers. We further experiment with dense retrievers explicitly trained to improve robustness against questions with typos since they improve robustness against ASR noise as well [144].

We need to underline that contrary to Sidiropoulos et al. [144], who assumed that the clean textual version of the spoken question is provided for training the retrievers, we follow a real-world scenario where only spoken questions are available; thus, we need to transcribe them for training the retrievers. Our experimental results showed that we build stronger baselines by training on the transcriptions of spoken questions. In particular, we experiment with the following retrievers:

- **BM25** is a traditional retriever based on term-matching. Question and retrieved passages have lexical overlap.

- **Dense Retriever (DR)** is used for scoring question-passage pairs and consists of two separate neural networks (dual-encoder), each representing a question and a passage. Given a question, a positive passage, and a set of negative passages, the learning task trains the two encoders by minimizing a loss function, typically softmax cross-entropy, to encourage positive question-passage pairs to have smaller distances than the negative ones. To train the *DR* model, we use the training scheme and hyperparameters described in [73] with a batch size of 64 which is the largest we can fit in our GPU setup.

- **Dense retriever with data augmentation (DR+Augm.)** alternates the training scheme of classic dense retrieval via the addition of data augmentation. Recently, there were works that explored a data augmentation approach for robustifying dense retrievers against typoed questions [139, 196]. Specifically, during the training phase of the dense retriever, an unbiased coin is drawn for each question that appears. If tails, the unchanged question is used for training. Otherwise, the question is injected with typos. Typos are sampled uniformly from one of the available typo generators (e.g., random insertion/deletion/substitution, neighbor swapping).

The *DR+Augm.* model we use in our experiments is trained following the training process described in [139]. The hyperparameters of the dense retriever remain the same as in the *DR* case.

- **Dense retriever with CharacterBERT and self-teaching (CharacterBERT-DR+ST)** [197] extends *DR+Augm.* by (i) incorporating CharacterBERT, which drops the WordPiece tokenizer and replaces it with a CharacterCNN module, and (ii) by adding a loss that distills knowledge from questions without typos into the questions with typos. The distillation loss forces the retrieval model to generate similar rankings for the original question and its typoed variations. That is achieved by minimizing the KL-divergence:

$$\mathcal{L}_{KL} = \tilde{s}(q', p) \cdot \log \frac{\tilde{s}(q', p)}{\tilde{s}(q, p)}, \tag{6.5}$$

where q and q' represent the original question and its typoed variant, respectively while $\tilde{s}$ is the softmax normalized similarity score. The $\mathcal{L}_{KL}$ loss (Equation 6.5) is combined with the supervised softmax cross-entropy loss $\mathcal{L}_{CE}$ to form the final loss:

$$\mathcal{L}_{ST} = \mathcal{L}_{KL} + \mathcal{L}_{CE}. \tag{6.6}$$

For training *CharacterBERT-DR+ST*, we follow the original work by Zhuang and Zuccon [197].

6.4.2 Datasets and evaluation

We conduct our experiments on the spoken versions of MSMARCO passage ranking [113], and Natural Questions [78] introduced by Sidiropoulos et al. [144]. In the Spoken-MSMARCO and Spoken-NQ, the question is in spoken form, while the passage is in the form of text. For the former dataset, the underlying corpus consists of 8.8 million passages, $\sim 400K$ training samples, and $6,980$ development samples, with an average sample duration of $3sec$ ($\sim$6 words). Spoken-NQ facilitates $58,880$ training, $6,515$ development, and $3,610$ test samples with an average sample duration of $3.86sec$ ($\sim$9 words). Questions can be answered over Wikipedia. Concerning the pipeline approaches that employ ASR for the transcription of the spoken questions, the WER for Spoken-NQ (test) is 20.10% and for Spoken-MSMARCO (dev) it is 32.87%.

To measure the retrieval effectiveness of the models on Spoken-MSMARCO, we use the official metric of the original MSMARCO dataset, namely, Mean Reciprocal Rank ($MRR@10$) and the commonly reported Recall ($R@50, R@1000$). Similar to the original MSMARCO, we report the metrics on the development set, since the ground-truths for the test set are not publicly available. Following previous works on NQ and Spoken-NQ, we use answer recall at the top-k ($AR@20, AR@100$) retrieved passages. Answer recall measures whether at least one of the top-k retrieved passages contains the ground-truth answer string, then $AR@k = 1$ otherwise $AR@k = 0$.

6.4.3 Implementation details

We train our multimodal dense retriever using the in-batch negative setting described in [73]; with one hard negative passage per question and a batch size of 64, the largest batch

Table 6.1: Retrieval results on Spoken-NQ and Spoken-MSMARCO. End2end "✗" accounts for *ASR-Retriever* pipeline approaches that can not be trained in an end-to-end manner. Training Questions "ASR" indicates that the transcriptions of spoken questions are used.

	Training Questions	End2end	Spoken-NQ (test)		Spoken-MSMARCO (dev)		
			AR@20	AR@100	R@50	R@1000	MRR@10
BM25	-	✗	36.98	52.49	24.71	45.34	6.97
DR	ASR	✗	68.36	78.53	52.06	79.35	17.74
DR+Augm.	ASR & Typos	✗	**69.77**	80.05	52.97	80.56	17.87
CharacterBERT-DR+ST	ASR & Typos	✗	68.22	**80.3**	**53.88**	78.95	**20.53**
Multimodal DR (Ours)	Speech	✓	57.25	70.11	51.37	**81.34**	15.77

we could fit in our GPU setup. The question and the passage encoders are implemented by *HuBERT-Base* [66] and *BERT-Base* [32] networks, respectively. We take the first embedding from the two output representations of each sequence to represent their corresponding speech and text sequences. Additionally, our experimental results showed that having different learning rates for the question and passage encoders leads to more effective training. Specifically, we set the learning rate to $2e$-4 for the question encoder and $2e$-5 for the passage encoder. To this end, we train with the Adam optimizer and linear scheduling with 0.1 warm-up for up to 80 epochs for the small Spoken-NQ dataset and 10 for the larger Spoken-MSMARCO dataset.

To end up with the abovementioned parameters, we searched learning rates ranging from $2e$-6 to $2e$-4 (for cases where question and passage encoders share the same learning rate or have different ones) and explored *HuBERT-Base* and *Wav2Vec2-Base* question encoders. We also experiment with warming up the question embedding space before training our retriever, following the sequence-level alignment approach described by Chung et al. [22]. However, we did not see any improvements. We chose the best hyperparameters with respect to the best average rank in the development split of Spoken-NQ [73].

6.5 Results & Discussion

6.5.1 Main results

To answer **RQ5.1**, we compare the retrieval performance of our multimodal dense retriever against the *ASR-Retriever* pipelines we described in Section 5.2.4. From Table 6.1 we note that our model is highly competitive on the Spoken-MSMARCO dataset, while the pipeline approaches perform significantly better on the Spoken-NQ dataset. This discrepancy is twofold. First, our multimodal dense retriever performs better on shorter questions. We conjecture that the low performance of our model on longer questions is due to encoding the spoken question into a single vector which might not be enough to capture the necessary information as the length of the question increases. Second, the higher the word error rate the higher the negative impact on the *ASR-Retriever* pipelines. Spoken-MSMARCO has shorter questions and a higher word

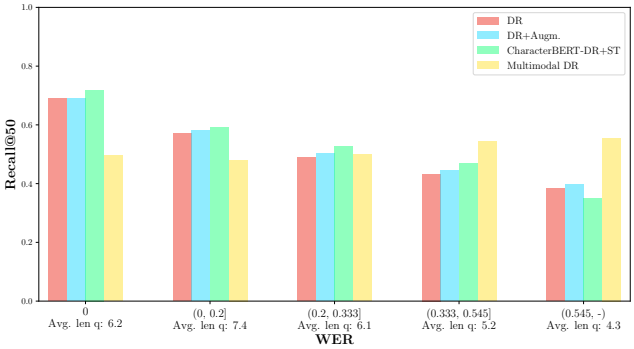


Figure 6.2: Retrieval performance w.r.t. the WER of questions; on Spoken-MSMARCO (dev). Each bin with a non-zero WER has $\sim$1300 samples, while the one with a zero WER has $\sim$1600 samples. We also report the average question length, in tokens, per bin.

error rate than Spoken-NQ (Section 8.5.1).

In Figure 6.2, we verify our aforementioned claims. Figure 6.2 reports the retrieval performance of the *ASR-Retriever* pipelines concerning the different word error rates of the ASR model and where our ASR-free method stands. Pipelines show strong results when no corrupted words are in the transcriptions (WER is 0). However, there is a significant drop in the retrieval performance of all the pipeline approaches as the word error rate increases. On the other hand, our ASR-free, multimodal dense retriever is significantly more stable across the different settings. At the same time, we see an increase in the performance of our approach as the length of the question decreases (see the average length under each bin in Figure 6.2). We conclude that adopting our multimodal dense retriever as the word error rate of the ASR model increases yields better results.

Despite the strong performance that *ASR-Retriever* pipelines achieve, there are several limitations we need to highlight. In Table 6.2, we compare our model to *ASR-Retriever* pipelines in terms of query time, the need for annotated speech data, and parameters. A major constraint for pipelines is the requirement in annotated speech for training an ASR model. In the real world, such data are not always in abundance. Annotated speech can be hard to obtain when dealing with low-resource languages or specialized domains such as the medical domain, where a general-purpose ASR system will underperform. In such scenarios, the applicability of *ASR-Retriever* pipelines is limited. In contrast, our approach is ASR-free, hence, does not need annotated speech. Regarding query time, as shown in Table 6.2, the query time of our model is substantially shorter than that of the *ASR-Retriever*. Passing the spoken question through an ASR model to obtain its transcription introduces additional overhead.

In our work, we are interested in the cases where ASR generates transcriptions with a higher word error rate; therefore, we conduct extensive analyses focusing on such cases.

Table 6.2: Comparison of our *Multimodal DR* and *ASR-Retriever* pipeline w.r.t. needs in annotated speech data, the number of model parameters, and query time.

	Annotated Speech	ASR #params	Retriever #params	Time
DR+Augm.	960h	95M	220M	45.3ms
CharacterBert-DR+ST	960h	95M	210M	42.5ms
Multimodal DR (Ours)	-	-	**200M**	**21.9ms**

Table 6.3: Retrieval performance on Spoken-MSMARCO (dev) for the cases where at inference time (i) the question has a mistranscription that the model encountered during training (Seen), or (ii) has a mistranscription that the model never encountered during training (Unseen). Unseen covers 1,883 samples, while Seen 3,437.

	Training Questions	End2end	Unseen			Seen		
			MRR@10	R@50	R@1000	MRR@10	R@50	R@1000
DR	ASR	✗	13.45	40.29	67.29	16.69	50.32	79.55
DR+Augm.	ASR & Typos	✗	13.41	41.08	69.63	17.18	51.75	80.83
CharacterBERT-DR+ST	ASR & Typos	✗	15.15	40.86	67.27	**19.75**	**52.32**	78.62
Multimodal DR (Ours)	Speech	✓	**18.52**	**54.26**	**82.11**	15.47	50.64	**81.58**

6.5.2 Analysis

Our multimodal dense retriever is ASR-free. Thus, there are no ASR errors that can propagate to the retriever. In contrast, in the *ASR-Retriever* pipeline, the transcribed question that arrives as input to the retriever will often contain mistranscribed words. Nevertheless, not every word in a question is of equal importance. Let us take as examples the following two transcriptions: "what channel is young sheldon on" → "what channel is young shelternon" and "who took the first steps on the moon in 1969" → "he took the first steps on the moon in nineteen sixty nine". Concerning the former, the corruption can lead the retrieval far from the main entity "young sheldon", while in the latter case, the error will have a limited impact on the retrieval. We claim that mistranscribing an important word can hurt retrieval performance more than mistranscribing less important ones. To verify our claim, we explore how the importance of the mistranscribed word impacts the retrieval performance of pipelines and how it compares against our ASR-free multimodal dense retriever (**RQ5.2**).

For our experiments, we define the relevant importance of a word in a question as the ratio of its IDF to the sum of the IDFs of every word in the question [139]. Figure 6.3 shows that as the importance of the mistranscribed word increases, there is a dramatic drop in the retrieval performance of the *ASR-Retriever* pipelines. At the same time, as more significant words are corrupted due to the failure of the ASR model, following our ASR-free multimodal dense retriever method is a promising alternative.

Our ASR-free multimodal dense retriever is trained on spoken questions. In contrast, in an *ASR-Retriever* pipeline, the retrieval model is trained on ASR transcriptions. As a result, the retriever encounters ASR mistranscription during training, similar to

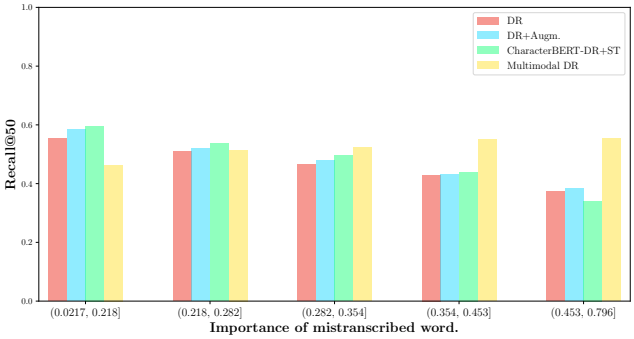


Figure 6.3: Retrieval results w.r.t. the relevant importance of the mistranscribed words; on Spoken-MSMARCO (dev). For questions with multiple mistranscribed words, we use the word with the highest relevant importance to assign the question to a bin. Bins have $\sim$1000 samples.

Table 6.4: Retrieval results on Spoken-MSMARCO (dev) for the setting where the dense retrievers of *ASR-Retriever* are not explicitly trained on ASR corrupted words.

	Training Questions	End2end	R@1000	MRR@10
BM25	-	✗	45.34	6.97
DR	Clean	✗	56.83	11.36
DR+Augm.	Clean & Typos	✗	64.57	13.02
CharacterBERT-DR+ST	Clean & Typos	✗	66.94	15.75
Multimodal DR (Ours)	Speech	✓	**81.34**	**15.77**

the ones it encounters during inference. However, a mistranscription of a particular word is strongly connected to its context. For instance, we can have the following two mistranscriptions for the word "exxon": "when did standard oil of new jersey become exxon" $\rightarrow$ "when did standard oil of new jersey become exon" and "where are exxon's refineries located" $\rightarrow$ "where a rexons refineries located".

For **RQ5.3**, we study the effectiveness of our ASR-free multimodal dense retriever against *ASR-Retriever* pipelines when the latter encounters previously unseen ASR mistranscribed words. We explore cases where a particular corrupted word during inference time was not part of the training set. In Table 6.3 we compare the retrieval performance for the cases of (i) previously seen and (ii) previously unseen ASR corrupted words.

Table 6.3 unveils that a big part of the strong performance we observe on the *ASR-Retriever* pipelines stems from the fact that retrievers are trained on the exact same mistranscriptions they encounter during inference. There is a decrease of more than 10 points in Recall for all pipelines when they deal with corruption due to ASR that was not part of the training set. Additionally, our multimodal dense retriever significantly outperforms all the pipelines under the unseen scenario.

Table 6.5: Comparison of different training schemes on Spoken-NQ. We indicate the learning rate of the question and passage encoder as q and p, respectively.

Pooling	Learning Rate	AR@20	AR@100
first	p: $2e$-5, q: $2e$-5	50.77	64.48
first	p: $2e$-5, q: $2e$-4	**57.25**	**70.11**
mean	p: $2e$-5, q: $2e$-4	56.59	69.97
max	p: $2e$-5, q: $2e$-4	53.57	67.45

Inspired by the results in Table 6.3, we set to study an extreme case of unseen ASR noise. In particular, we study how much the retrieval performance of pipeline approaches deteriorates when their dense retrievers are not explicitly trained on ASR noise (transcriptions). To do so, we train from scratch all the dense retrievers on clean questions instead of transcriptions. We report the results in Table 6.4. By comparing the retrieval models when they are explicitly aware of ASR corrupted words during training (as shown in Table 6.1) vs. when they are not (Table 6.4), we see a dramatic drop in Recall of more than 12 points for all *ASR-Retriever* pipelines; and as high as 23 points in the case of *DR*. From Table 6.4, we can further conclude that for the extreme case where the *ASR-Retriever* pipeline is not trained on ASR noise, following our multimodal dense retriever approach is necessary.

6.5.3 Ablation study on model training

To better understand how different model training schemes affect retrieval performance (**RQ5.4**), we perform an ablation on our multimodal dense retriever and discuss our findings below.

Learning Rate In traditional dense text retrieval, the same learning rate is used to update all the weights in the retriever [73, 124]. However, in our multimodal dense retrieval setup, the HuBERT (question encoder) and BERT (passage encoder) models are pre-trained independently to allow the usage of available large-scale unsupervised data. Therefore, there can be disparities between the two modalities that can hurt performance. To alleviate this problem we follow an alternative setting where the two encoders have different learning rates. In particular, since the language model contains more information than the speech model, we increase the learning rate of the passage encoder by a factor of 10. Comparing the values in the first block of Table 6.5, we find that the choice of learning rate is important for effectively training our multimodal dense retriever.

Pooling The next ablation involves different pooling methods for encoding the spoken question into a single vector. Following previous works on dense retrieval, we use the *[CLS]* token embedding output from BERT to encode the text passage [73]. In contrast, this decision is not that straightforward in the case of the spoken question. The HuBERT speech transformer we use for encoding the spoken questions does not

have a next-sentence prediction pre-training task as in BERT. Thus, there is no *[CLS]* token available. We assess different pooling strategies for encoding the spoken question, namely, max and mean pooling or taking the first embedding of the sequence as a pooling strategy. As we can see in Table 6.5, using the first token embedding output from HuBERT to represent the speech utterance leads to the best results.

For our ablation study, we reported results on the test split of Spoken-NQ (Table 6.5). However, we want to clarify that the decision for our best multimodal retriever was based on the development set, as discussed in Section 5.2.5.

6.6 Conclusions

In this chapter, we thoroughly analyzed the behavior of ASR-Retriever pipelines for passage retrieval for speech-based open-domain QA, showing their limitations, and we further introduced the first end-to-end trained, ASR-free multimodal dense retriever in order to tackle these limitations. Our experimental results showed that our multimodal dense retriever is a promising alternative to the *ASR-Retriever* pipelines on shorter questions and under higher word error rate scenarios. Furthermore, we unveiled that *ASR-Retriever* pipelines show a dramatic performance decrease as the word error rate of the ASR model increases, when important words in the spoken question are mistranscribed, and when dealing with mistranscriptions that have not been encountered during training. We showcased that our ASR-free multimodal dense retriever can overcome the aforementioned issues. To conclude, despite the limited performance of our proposed method on longer questions, we believe that our thorough analysis can spur community interest in passage retrieval for speech-based open-domain QA.

6.7 Limitations

In this chapter, all the models are trained using the hard negatives provided by the original datasets or mined from BM25 [73] and by employing the base versions of the speech and language transformer models (see Section 6.4.3). Therefore, we leave the application of more complex hard negatives mining techniques, such as mining from the dense index of a previous checkpoint of the dense retriever itself [174], and larger models (e.g., BERT-Large and HuBERT-Large) to future work.

As we saw in Section 6.5.1, our multimodal dense retriever showed promising results against its *ASR-Retriever* counterparts on shorter questions under high word error rate scenarios. We conjecture that the limited performance of our approach on long questions is due to encoding all the information from the speech signal in a single vector, and we leave exploring a multi-vector retrieval approach as future work. *ASR-Retriever* pipelines can produce significantly better results compared to our method for cases where the ASR model can perform high-quality transcriptions with low word error rate. We did not experiment with more advanced ASR models, such as the recently proposed Whisper [125], since such models not only require an abundance of annotated speech (i.e., $680,000$ hours) but also our goal in this chapter was to provide alternatives for cases with higher word error rates in the transcriptions.

This chapter concludes the second part of the thesis, which focuses on improving the robustness of speech-based search with neural retrieval and aims to support multimodality in search engines. In Chapter 7, we move to a different research theme and explore how to improve the efficiency of training neural retrievers on low resources for multi-hop retrieval, thereby helping search engines deal with complex queries.

Part III

Improving the Efficiency of Training Neural Retrievers on Low Resources for Multi-hop Retrieval

7

Computationally Efficient Hybrid Retrieval for Answering Complex Queries

The information needs of real users do not only take the form of simple queries where the answer is explicit in a single document. Real users may have complex information needs, which they express as complex queries, necessitating multi-hop retrieval to aggregate information from multiple documents to infer the answer. Dense retrieval has been successfully applied to complex queries. However, training an effective dense retriever for multi-hop retrieval is computationally expensive. To address this challenge, we set out to answer **RQ6**: Can we train an effective dense retriever for multi-hop retrieval with limited computing resources?

In the scope of this chapter, we use "retrieval for multi-hop QA" and "retrieval for multi-hop queries" as an alias for "multi-hop retrieval."

7.1 Introduction

The second Strategic Workshop on Information Retrieval (IR) prioritized moving beyond simple document ranking towards more realistic settings required to address complex information needs [1]. While supporting complex information needs by decomposing them into independent facets, subtopics or sub-tasks has already received a lot of attention within the IR community [33, 72], the task of multi-hop (complex) question answering (QA) remains largely overlooked from an IR point of view.

Multi-hop QA requires retrieval and reasoning over multiple pieces of information [179]. For instance, consider the multi-hop question: "Where is the multinational company founded by Robert Smith headquartered?" To answer this question we first need to retrieve the passage about Robert Smith in order to find the name of the company he founded (General Mills), and subsequently retrieve the passage about General Mills, which contains the answer to the question (Golden Valley, Minnesota). Even though multi-hop QA requires multiple retrieval hops, it is fundamentally different from session search [83, 176] and conversational search [30, 158, 163], since in multi-hop QA the

This chapter was published as G. Sidiropoulos, N. Voskarides, S. Vakulenko, and E. Kanoulas. Combining lexical and dense retrieval for computationally efficient multi-hop question answering. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing, SustaiNLP@EMNLP 2021*. Association for Computational Linguistics, 2021.

information need of the user is expressed in a single question, thus not requiring multiple turns of interaction.

QA systems typically consist of (i) a retriever that identifies the passage/document in the underlying collection that contains the answer to the user's question, and (ii) a reader that extracts or generates the answer from the identified passage [19]. Given that often the answer cannot be found in the top-ranked passage, inference follows a standard beam-search procedure, where top-k passages are retrieved and the reader scores are computed for all k passages [82]. However, readers are very sensitive to noise in the top-k passages, thus making the performance of the retriever critical for the performance of QA systems [178]. This is further amplified in multi-hop QA, where multiple retrieval hops are performed; potential retrieval errors get propagated across hops and thus severely harm QA performance.

The majority of current approaches to multi-hop QA use either traditional IR methods (TF-IDF, BM25) [123] or graph-based methods for the retriever [2, 114]. However, those approaches have serious limitations. The former approaches require high lexical overlap between questions and relevant passages, while the latter rely on an interlinked underlying corpus, which is not always available. Recently, Xiong et al. [175] introduced a dense multi-hop passage retrieval model that constructs a new query representation based on the question and previously retrieved passages and subsequently uses the new representation to retrieve the next set of relevant passages. This model achieved state-of-the-art results without relying on an interlinked underlying corpus.

Even though dense retrieval models achieve state-of-the-art results on multi-hop QA, they are computationally intensive, requiring multiple GPUs to train. Existing work only reports results for the cases where such resources are available. It is not clear how feasible it is to train such models in a low-resource setting. In this chapter, we focus on developing an efficient retriever for multi-hop QA that can be trained effectively in a low computational resource setting. We break down **RQ6** into the following research sub-questions:

RQ6.1 How does the performance of dense retrieval compare to lexical and hybrid approaches?

RQ6.2 How does the performance degrade in low computational resource settings?

RQ6.3 How do different passage retrieval approaches perform across hops?

Our main contributions are the following: (i) we propose a hybrid (lexical and dense) retrieval model that is competitive against its fully dense competitors while requiring eight times less computational power, (ii) we perform a thorough analysis of the performance of dense passage retrieval models on the task of multi-hop QA, and (iii) we conduct a qualitative analysis to get further insights on the working of the models.[1]

[1] Our trained models and data are available at `https://github.com/GSidiropoulos/hybrid_retrieval_for_efficient_qa`.

7.2 Task Description

Let $p \in \mathcal{C}$ denote a passage within a passage collection $\mathcal{C}$, and q a multi-hop question. Given q and $\mathcal{C}$ the task is to retrieve a set of relevant passages $\mathcal{P} = \{p_1, p_2, \ldots, p_n\}$, where $p_i \in \mathcal{C}$. In the multi-hop scenario we consider here, not all relevant passages can be retrieved using the input question q alone. This is due to the fact that there is a low lexical overlap or semantic relationship between question q and one or more of the relevant passages in $\mathcal{P}$. In this case, information from one of the relevant passages p_i is needed to retrieve another relevant passage p_j, where p_j may be lexically/semantically different from question q.

7.3 Related Work

The majority of open-domain QA methods use a pipeline approach that consists of a retriever and a reader component [2, 19, 34, 73, 81, 105, 114, 123]. While there has been a lot of attention on improving the reader component [34, 105], we focus on related work on the retriever part, as it is more relevant to our work in this chapter.

In the case of simple open-domain QA (single-hop) the retriever uses traditional term-based IR methods [19], often in combination with additional reranking steps [81]. Differently, recent dense passage retrieval methods use dense question and passage representations to capture the semantic similarity between question and relevant passages [73]. The aforementioned approaches often fail in multi-hop QA since more than one passages must be retrieved, with one (or more) of them having little lexical or semantic similarity to the question. In multi-hop QA, many works follow graph-based approaches, constructing a document graph that employs hyperlinks in the underlying corpus, either explicitly through a Wikipedia-like hyperlink structure or implicitly via entity linking [2, 114]. However, these methods are limited by the availability of hyperlinks and the entity linking performance. On the other hand, iterative passage retrieval approaches with multiple rounds of term-based retrieval are limited by the lexical retrieval per se [123]. Recently, dense passage retrieval methods were introduced to multi-hop QA [175] questions and previously retrieved passages encoded to retrieve the next relevant set of passages. Nonetheless, in many cases, they do not properly exploit the lexical overlap between questions and passages [73].

7.4 Method

In this section we discuss the preliminaries and then introduce our hybrid retrieval approach.

7.4.1 Preliminaries

Rerank Multi-stage ranking architectures can provide a balance between model complexity and search latency by subsequent stages that re-rank the set of candidates that is passed. The initial retrieval stage uses traditional lexical methods over the whole corpus $\mathcal{C}$. In detail, it produces a candidate set C_0 by treating the given question q as a

"bag of words" and ranking the passages using a standard inverted index. The next stage is a re-ranking stage that estimates the relevance of each candidate passage $p_i \in C_0$ to the question q. For this purpose, it employs a pointwise re-ranker, which is a BERT model used as a binary relevance classifier.

Dense Passage Retrieval In contrast to term-based retrieval that matches sparse vectors, dense retrieval matches dense vector representations. It uses encoders (e.g., BERT) to map q and p into dense vectors $E_Q(q)$ and $E_P(p)$, respectively, and computes their similarity:

$$sim(q, p) = E_Q(q)^T E_P(p).$$

At training time, given a question q, a relevant passage p^+, and a set of irrelevant passages $\{p_1^-, p_2^-, \ldots, p_m^-\}$, the model learns to rank p^+ higher via the optimization of the negative log-likelihood of the relevant passage. At inference time, passages can be efficiently retrieved via approximate nearest neighbor search over the maximum inner product with the question.

7.4.2 Hybrid retrieval

Our system solves the multi-hop QA problem in an iterative fashion. In detail, the retrieval of a passage at hop t is conditioned on the previously retrieved passages (up to $t-1$) and the given question:

$$P(\mathcal{P}|q) = \prod_{t=1}^{n} (q_t|q, p_{t_1}, \ldots, p_{t-1}).$$

For the retrieval of the passage at $t = 1$ we condition only on q and we rely on the abovementioned Rerank system. For the subsequent retrieval steps, we rely on DPR. We build a new representation for the question based on the previously retrieved passages and the initial question and use this new representation to retrieve passages. Specifically, we concatenate the question and the previously retrieved passages and feed them to DPR.

7.5 Experimental Setup

In this section, we describe the dataset used in our experiments, the metrics we use to answer our research questions and the models we compare against.

7.5.1 Dataset

For our experiments we focus on the HotpotQA dataset and particularly the full-wiki setting [179]. HotpotQA is a large-scale 2-hop QA dataset where in contrast with other multi-hop QA datasets [151], questions are not limited to knowledge graphs, but instead answers to questions must be found in the context of the entire Wikipedia. Questions in HotpotQA fall into one of the following categories: bridge or comparison. In *bridge* questions, the bridge entity that connects the two relevant passages is missing,

Table 7.1: Dataset statistics.

Split	Type	# Samples	%
Train	Bridge	72,991	80.7
	Comparison	17,456	19.3
	All	90,447	100.0
Dev	Bridge	5,918	79.9
	Comparison	1,487	20.1
	All	7,405	100.0

while in *comparison* questions the main two entities (of the two relevant passages) are both mentioned and compared. For instance, "When did the show that Skeet Ulrich is currently starring in premiere?" is a bridge question where the bridge entity "Riverdale (2017 TV series)" is missing. On the other hand, "Which has smaller flowers, Campsis or Kalmiopsis?" is a comparison question. Table 7.1 provides the dataset statistics. We use the train split for training our supervised models and report results on the development set.

7.5.2 Metrics

Following previous work [175, 179], we report passage Exact Match (EM) to measure the overall retrieval performance. Exact Match is a metric that evaluates whether both of the ground-truth passages for each question are included in the retrieved passages (then $\mathrm{EM} = 1$ otherwise $\mathrm{EM} = 0$). Note that metrics such as EM and F1 w.r.t. a question's answer (Ans) and supporting facts on sentence-level (Sup) do not fit in our experimental setup since we focus on the retrieval part of the pipeline and not on the reading. We also report on Recall (R) and Mean Reciprocal Rank (MRR) for the per-hop retrieval evaluation.

7.5.3 Models

In this section, we describe the models we experiment with.

Single-hop models

Given a question, single-hop models retrieve a ranked list of passages. Thus, they are not aware of the multi-hop nature of the task. We experiment with the following single-hop models:

- **BM25** is a standard lexical retrieval model. We use the default Anserini parameters [177].
- **Rerank** is a standard two-stage retrieve-and-rerank model that first retrieves passages with BM25 and then uses BERT to rerank the top-k passages [115]. The BERT (base) classifier was trained with a point-wise loss [115]. It was fine-tuned on the train split

of HotpotQA for 2 epochs. Training took 5 hours with batch size of 8 using a single 12GB GPU. We experimented with $k = 100$ and $k = 1000$, and found that $k = 100$ results in a better reranking performance at the top positions.

- **DPR** is a dense passage retrieval model for simple questions [73]. Given a question q, a relevant passage p^+ and a set of irrelevant passages $\{p_1^-, p_2^-, \ldots, p_m^-\}$, the model learns to rank p^+ higher via the optimization of the negative log-likelihood of the relevant passage. To train DPR on HotpotQA, a multi-hop QA dataset, we follow the procedure described in [175]. This model was trained for 25 epochs ($\sim$2 days) on a single 12GB GPU, using a RoBERTa-based encoder.

Multi-hop models

These models are aware of the multi-hop nature of the task. They recursively retrieve new information at each hop by conditioning the question on information retrieved on previous hops [175]. In practice, at each hop t the question q and the passage retrieved in the previous hop p_{t-1} get encoded as the new query $q_t = h(q, p_{t-1})$, where $h(\cdot)$ is the question encoder, to retrieve the next relevant passage; when $t = 1$ then we have just the question. Differently from single-hop models, at inference time, given a question, beam search is used to obtain the top-k passage pair candidates. The candidates to beam search at each hop are generated by a similarity function using the query representation at hop t, and the beams are scored by the sum of the individual similarity scores. We experiment with the following multi-hop models:

- **MDR** is a state-of-the-art dense retriever for multi-hop questions [175]. It extends DPR in an iterative fashion by encoding the question and passages retrieved in previous hops as the new query to retrieve the next relevant passages. This model was trained for 25 epochs ($\sim$3 days) on a single 12GB GPU, using a RoBERTa-based encoder, without the memory bank mechanism [172]. The memory bank mechanism is dropped since it is very expensive to compute and its contribution to retrieval performance is limited.

- **MDR (full)** is MDR with the additional memory bank mechanism, trained for 50 epochs on 8$\times$32GB GPUs by Xiong et al. [175].

- **Rerank + DPR$_2$** is a hybrid model we propose in this chapter. Specifically, for the first hop we rely on the BERT-based re-ranking model described in Section 7.5.3 (Rerank), while for the second hop we train a DPR only on second hop questions (DPR$_2$). To train the latter, we build a variation of HotpotQA where the question gets concatenated with the ground truth passage of the first hop, and the second hop ground truth passage is the only relevant passage to be retrieved. DPR$_2$ was trained for 25 epochs ($\sim$1 day) on a single 12GB GPU, using a RoBERTa-based encoder.

7.6 Results

In this section, we present our experimental results that answer our research questions.

Table 7.2: Overall retrieval performance. In the first group we show single-hop models, while in the second group we show multi-hop models.

Model	EM@2	EM@10	EM@20
BM25	0.127	0.320	0.395
Rerank	0.314	0.476	0.517
DPR	0.116	0.275	0.336
MDR	0.440	0.581	0.619
Rerank+DPR$_2$	0.599	0.732	0.762
MDR (full)	**0.677**	**0.772**	**0.793**

7.6.1 Overall performance

Here we aim to answer **RQ6.1** by comparing the retrieval performance of the models we consider. In Table 7.2, we observe that the single-hop models perform much worse than the multi-hop models. This is expected since single-hop models are not aware of the multi-hop nature of the task.

As for the multi-hop models, we observe that MDR (full) achieves the best performance at the higher positions in the ranking. It is important to underline here that MDR (full) uses considerably more resources than Rerank+DPR$_2$ and MDR. The latter two use relatively limited computational resources and a comparison between them is more fair than comparing against MDR (full) (see Section 7.5.3).[2] We observe that our Rerank+DPR$_2$ outperforms MDR on all metrics while it is also competitive against MDR (full), especially w.r.t. EM@10 and EM@20. This is due to the fact that often questions and their relevant passages are not only semantically related, but also have high lexical overlap. This is also highlighted by Karpukhin et al. [73], who reported that dense retrieval has performance issues when the question has high lexical overlap with the passages.

7.6.2 Performance for limited resources

In this section, we answer **RQ6.2** by comparing the retrieval performance of MDR and DPR as provided in [175], against the same models trained with limited computational resources.

In Table 7.3 we see that performance drops significantly as we limit resources for both DPR and MDR. This is a result of the training scheme that is used in [73] and [175]. More specifically, DPR and MDR rely on using in-batch negatives both for decreasing the training time (positive passages of a question are reused as negative passages for the rest of the questions in the batch, instead of having to sample new ones beforehand), and for improving accuracy (bigger batch size will produce more in-batch negatives, thus increasing the number of training samples). When we have limited resources, training

[2]Even though the memory bank mechanism is omitted from MDR, the comparison of Rerank+DPR$_2$ and MDR remains fair since this particular mechanism can also be potentially applied to Rerank+DPR$_2$ (in the DPR part).

Table 7.3: Analysis of how computational resources affect the performance of MDR and DPR. The MDR (full) configuration is provided by [175]. A different beam size can slightly change the results from what was originally reported. MDR and DPR, both trained on 8 GPUs, are not available and therefore we report the results as they were reported in [175].

Model	Encoder	# GPU	# Epochs	Batch	# Gradient acc. steps	EM@2	EM@10	EM@20
MDR (full)	RoBERTa	8 × 32GB	50	150	1	0.677	0.772	0.793
MDR	RoBERTa	8 × 32GB	50	150	1	0.637	0.742	0.772
	RoBERTa	4 × 24GB	50	28	1	0.606	0.711	0.735
	RoBERTa	4 × 24GB	25	28	1	0.550	0.668	0.698
	RoBERTa	4 × 24GB	20	28	1	0.537	0.659	0.687
	RoBERTa	1 × 12GB	25	4	32	0.440	0.581	0.619
	BERT	1 × 12GB	25	4	32	0.421	0.560	0.597
DPR	RoBERTa	8 × 32GB	50	256	1	0.252	0.454	0.521
	RoBERTa	4 × 24GB	25	128	1	0.223	0.427	0.487
	RoBERTa	1 × 12GB	25	8	32	0.116	0.275	0.336

time gets significantly longer (since we use fewer GPUs), and therefore we have to compromise for fewer training epochs while the batch size is restricted by the GPU memory size. For instance, training for 50 epochs takes $\sim$1 day on 8$\times$32GB GPUs, while it takes $\sim$6 days on a single 12GB GPU.

In addition, when comparing MDR trained on 4$\times$24GB GPUs against MDR trained on a single 12GB GPU, for 25 epochs each, we observe that even though we can simulate bigger batch sizes by using gradient accumulation,[3] we do not observe an increase in performance. This is a consequence of the fact that the number of in-batch negatives is limited by the real batch size. Note that we also observe a similar trend for DPR.

In summary, computational resources are of vital importance for multi-hop dense retrieval models. Hence, in the case where only limited resources are available, following a hybrid (lexical and dense) approach such as our proposed Rerank+DPR$_2$ seems to be a good choice. As we showed in Tables 7.2 and 7.3, Rerank+DPR$_2$ (trained on a single GPU) performs similarly to MDR trained on 4 GPUs and is competitive against MDR trained on 8 GPUs.

[3]Gradient accumulation is a mechanism that accumulates the gradients and the losses over consecutive batches for a specific number of steps (without updating the parameters), and then updates the parameters based on the cumulative gradient [109].

Table 7.4: Two example bridge questions for which Rerank+DPR$_2$ retrieves both relevant passages at the top positions while MDR fails to do so. Lexical overlap is indicated with underlined text while the answer to the question is highlighted.

	q **[bridge]: Who was the defense counsel of a German woman who underwent Catholic exorcism rites during the year before her death?**
Rerank+DPR$_2$	**Anneliese Michel**: Anneliese Michel] (21 September 1952 – 1 July 1976) was a German woman who underwent Catholic exorcism rites during the year before her death. Later investigation determined that she was malnourished and dehydrated. . . **Erich Schmidt-Leichner**: Erich Schmidt-Leichner (14 October 1910 – 17 March 1983) was a German lawyer who made a name as a distinguished defense counsel at the Nuremberg Trials (1945 - 1946). In 1978, he was a defense counsel in the "Klingenberg Case" (Anneliese Michel). . .
MDR	**Maria Pauer**: Maria Pauer (October 1734/36 – 1750 in Salzburg), was an alleged Austrian witch. She was the last person to be executed for witchcraft in Austria. **Franz Burkard (died 1539)**: Franz Burkard (died 1539) was a canon lawyer in Ingoldstadt who opposed Lutheranism, particularly in the trial of Andreas Seehofer.
	q **[bridge]: The astronomer who formulated the theory of stellar nucleosynthesis co-authored what landmark paper?**
Rerank+DPR$_2$	**Fred Hoyle**: Sir Fred Hoyle FRS (24 June 1915 – 20 August 2001) was an English astronomer who formulated the theory of stellar nucleosynthesis . . . **B2FH paper**: The BFH paper , named after the initials of the authors of the paper, Margaret Burbidge, Geoffrey Burbidge, William Fowler, and Fred Hoyle, is a landmark paper of stellar physics published in "Reviews of Modern Physics" in 1957 . . .
MDR	**Sequence hypothesis**: The sequence hypothesis was first formally proposed in the review "On Protein Synthesis" by Francis Crick in 1958 . . . **Francis Crick**: Francis Harry Compton Crick (8 June 1916 – 28 July 2004) was a British molecular biologist, biophysicist, and neuroscientist, most noted for being a co-discoverer of the structure of the DNA molecule in . . .

Table 7.5: Two example bridge questions for which MDR retrieves both relevant passages at the top positions while Rerank+DPR$_2$ fails to do so. The answer to the question is highlighted.

	q **[bridge]: What Golden Globe Award actor starred in the film Little Fugitive?**
Rerank+DPR$_2$	**Ali MacGraw:** Elizabeth Alice "Ali" MacGraw (born April 1, 1939) is an American actress, model, author, and animal rights activist. She first gained attention with her role in the 1969 film "Goodbye, Columbus", for which she won the Golden Globe Award for Most Promising Newcomer. She reached international fame in 1970's "Love Story", for which she was nominated for an Academy Award for Best Actress and won the Golden Globe Award for Best Actress in a Motion Picture – Drama ... **Little Fugitive**: Little Fugitive (1953) is an American film written and directed by Raymond Abrashkin (as "Ray Ashley"), Morris Engel and Ruth Orkin, that tells the story of a child alone in Coney Island.
MDR	**Little Fugitive (2006 film)**: Little Fugitive is a 2006 remake of the film of the same name. It was directed by Joanna Lipper and produced by Nicholas Paleologos. The film is set in present day Brooklyn and tells the story of 11-year-old Lenny (Nicolas Martí Salgado) who must take care of his 7-year-old brother, Joey (David Castro), while their father (Peter Dinklage) is in jail and their mother work long hours a nursing home ... **Peter Dinklage**: Peter Hayden Dinklage (, born June 11, 1969) is an American actor and film producer. He has received numerous accolades, including a Golden Globe Award and two Primetime Emmy Awards.
	q **[bridge]:Which American professional poker player also starred in the 2015 movie "Extraction"?**
Rerank+DPR$_2$	**Allen Cunningham**: Allen Cunningham (born March 28, 1977) is an American professional poker player who has won five World Series of Poker bracelets. **Extraction (film)**: Extraction is a 2015 American action-thriller film directed by Steven C. Miller and written by Umair Aleem. The film stars Kellan Lutz, Bruce Willis, Gina Carano, D. B. Sweeney, Dan Bilzerian and Steve Coulter ...
MDR	**Extraction (film)**: Extraction is a 2015 American action-thriller film directed by Steven C. Miller and written by Umair Aleem. The film stars Kellan Lutz, Bruce Willis, Gina Carano, D. B. Sweeney, Dan Bilzerian and Steve Coulter ... **Dan Bilzerian**: Dan Brandon Bilzerian (born December 7, 1980) is an American professional poker player.

7.6.3 Performance per hop

Previous work reports retrieval performance only at the end of the retrieval pipeline [114, 175]. However, since we are dealing with questions that require multi-hop reasoning, it is important to get insights into how the models under comparison behave at each hop.

In this section, we aim to answer **RQ6.3** by comparing the models based on their performance per hop. We follow Xiong et al. [175] to derive an order for the passages: the passage that includes the answer is considered to be the second passage. In order to measure the performance on the first hop, we take as input the question only and measure the model's ability to retrieve the two relevant passages. Specifically for MDR, which is a multi-hop model, we stop on the first iteration, before the use of beam search. In order to measure the performance on the second hop (independently of the performance of the first hop), we take as input both the question and the ground-truth passage of the first hop and measure the model's ability to retrieve the passage of the second hop.

First hop performance

Table 7.6 shows the first-hop performance of Rerank, DPR, and MDR. Rerank, which uses BM25 for the initial retrieval step, outperforms both MDR and DPR, which are purely dense models. This validates what we found in Section 7.6.1: lexical matching is important for this task. This is because questions may have large lexical overlap with either both relevant passages in the case of comparison questions or at least one passage in the case of bridge question (since the bridge entity is missing from the question). Table 7.6 also highlights that many samples in HotpotQA may not require multi-hop retrieval, since both passages can often be retrieved using the original question only: both relevant passages can be retrieved using the original question at top-10 for the majority of the comparison questions by the MDR model ($\mathrm{EM@10} = 0.825$) and for almost half of the bridge questions by the Rerank model ($\mathrm{EM@10} = 0.418$). The main reason behind this phenomenon are the lexical biases introduced by human annotators during the construction of the HotpotQA dataset. Specifically, annotators had to generate a question after being given two passages. The question had to be answerable by gathering evidence only from these two passages. Therefore, in many cases, there is a perhaps unrealistically high lexical overlap between the question and the passages. This can be avoided in the future by following an information-seeking approach where annotators generate a question for which they do not know the answer beforehand [23].

In Table 7.6 we also report performance for bridge and comparison questions separately. We can clearly see a dramatic decrease in performance for the case of bridge questions, compared to comparison questions, for all three models. The high performance of all the models for comparison questions is due to the fact that there is high lexical overlap and a strong semantic relationship between the question and both relevant passages. In contrast, for bridge questions where one of the relevant passages is both lexically and semantically distant from the initial question, we observe lower performance; especially for EM.

Table 7.6: Performance on the 1st hop for both passages.

	All					Bridge					Comparison				
Model	EM@2	R@2	EM@10	R@10	MRR	EM@2	R@2	EM@10	R@10	MRR	EM@2	R@2	EM@10	R@10	MRR
Rerank	**0.314**	**0.573**	**0.476**	**0.698**	**0.824**	**0.229**	**0.516**	**0.418**	**0.665**	**0.794**	**0.650**	**0.800**	0.705	0.830	**0.942**
DPR	0.116	0.351	0.275	0.495	0.581	0.064	0.288	0.179	0.416	0.512	0.324	0.603	0.657	0.808	0.855
MDR	0.137	0.463	0.239	0.550	0.777	0.041	0.394	0.091	0.461	0.738	0.521	0.736	**0.825**	**0.907**	0.931

Table 7.7: Performance on the 2nd hop. In this experiment, $R = EM$ since there is only one relevant passage to retrieve in the 2nd hop.

	All			Bridge			Comparison		
Model	R@2	R@10	MRR	R@2	R@10	MRR	R@2	R@10	MRR
Rerank	0.555	0.583	0.548	0.615	0.649	0.606	0.317	0.319	0.315
DPR_2	**0.817**	**0.899**	**0.792**	**0.783**	**0.876**	**0.753**	0.956	0.991	0.948
MDR	0.770	0.844	0.761	0.718	0.805	0.709	**0.975**	**0.997**	**0.970**

Second-hop performance

In Table 7.7 we see that DPR_2 outperforms MDR for the second hop. This is an outcome of training the former as a single-hop retriever but for the second hop; as mentioned in Section 7.5.3, given a question concatenated with the first relevant passage, the goal is to retrieve the second relevant passage. In contrast, MDR is trained for multi-hop retrieval. This shows that MDR still has room for improvement when it comes to retrieving the second relevant passage given the first relevant passage. Finally, Rerank shows the worst results, mainly because of the poor performance of BM25 when given the concatenation of a question and a passage as the query. Further analysis showed that one of the main reasons behind the low performance of Rerank on the comparison questions in the second hop is the low recall of the initial retrieval step (R@1000 for BM25 is only 0.604).

7.6.4 Error analysis

We perform a qualitative analysis to gain further insights into where the models succeed or fail. We compare specific cases where our hybrid model (Rerank+DPR_2) retrieves both relevant passages successfully while MDR fails and vice versa. For our analysis we focus on bridge questions since comparison questions are more straightforward to retrieve.

Table 7.4 shows two typical examples of questions for which Rerank+DPR_2 retrieves both relevant passages at the top positions while MDR fails to do so. When there is a high lexical overlap between the question and a relevant passage, our hybrid model can capture this exact n-gram match and improve the performance. In contrast, fully dense models seem incapable of capturing this. In particular, this lexical overlap can be between the question and both relevant passages for the case of comparison questions or between the question and the first relevant passage for bridge questions.

In bridge questions, if the lexical overlap is between the question and the second passage then our hybrid model favors passages in which this phrase appears, and therefore it retrieves an irrelevant first passage; leading to an irrelevant second passage as well. MDR on the other hand manages to retrieve both relevant passages at the top positions. Those are the cases where the lexical overlap is used in the given question in order to disambiguate the final answer. Examples can be found in Table 7.5.

In the first example, “Golden Globe Award” is used in the given question in order to disambiguate the final answer, since in the film “Little Fugitive” there is more than one

actor involved. Therefore, "Golden Globe Award" must be used to assist the retrieval of the second passage. Since Rerank+DPR$_2$ builds on top of BM25, it favors passages in which this phrase appears, and therefore it retrieves an irrelevant first passage leading to an irrelevant second passage as well. On the other hand, MDR manages to retrieve both relevant passages at the top positions. Similarly, for the second example, "American professional poker player" is used to specify the actor that starred in the "Extraction" movie, hence supporting the retrieval of the second relevant passage.

7.7 Conclusion

In this chapter, we provided insights into the performance of state-of-the-art dense passage retrieval for multi-hop questions. We showed that Rerank+DPR$_2$ (a hybrid model we proposed that combines sparse and dense retrieval) outperforms MDR (the state-of-the-art multi-hop dense passage retrieval model) in the low resource setting, and it is competitive with MDR in the setting where MDR uses considerably more computational resources. In addition, we provided insights into how single-hop and multi-hop models perform in the first and second hop separately. In the first hop, we found that models that rely on lexical matching for the initial retrieval step outperform purely dense models. In the second hop, we found that MDR is outperformed by DPR when the latter is trained to only retrieve the relevant passage in the second hop only (DPR$_2$). Finally, we highlighted that the performance of fully dense retrieval models is hurt when using limited computational resources.

For future work, we plan to build on our insights to improve the performance of multi-hop models by combining the strengths of lexical and dense retrieval. Also, we aim to develop less computationally expensive multi-hop retrieval models. DPR, MDR, and our hybrid approach (for the second hop) all use batch-wise contrastive loss. Consequently, these models could benefit from a larger batch size with more in-batch negatives. Thus, in resource-constrained environments, it may be advantageous not to rely on batch-wise contrastive loss during training; which can potentially lead to better performance when training with small batches compared to a batch-wise contrastive loss. Future research could explore methods to accomplish this, for instance, by employing knowledge distillation and a pairwise loss function like Margin-MSE [63]. Finally, as we notice from our analysis of the per-hop performance, the retrievers' performance varies depending on the type of question. Therefore, a future direction could explore how to build retrieval models that are robust across different question types.

In this chapter, we studied multi-hop retrieval under a low computational resource scenario, intending to help search engines handle complex queries. Next, in Chapter 8, we address a different research theme: improving domain adaptation in KGQA with neural retrieval to assist search engines in accessing structured knowledge.

7.8 Reflections

Since the research we presented in this chapter was conducted, several advancements have been made in the domain, both in terms of efficient training of dense retrievers and the application of dense retrievers in multi-hop retrieval.

Previous work on representation learning has demonstrated that learning high-quality representations benefits from training with a batch contrastive loss and a large number of negatives [18, 73, 75]. In practice, in-batch negative training is a time-efficient way to reuse the negative examples already in the batch rather than creating new ones. However, this method still conditions the loss of each example on all batch examples and requires fitting the entire large batch into GPU memory. Gao et al. [45] proposed a gradient caching technique that removes in-batch data dependency in encoder optimization. The proposed method produced the same gradient update as training with a large batch. The authors successfully applied their method to train an effective dense retriever for traditional ad-hoc retrieval (i.e., single-hop). That said, this approach can be applied to multi-hop retrieval in order to efficiently learn an effective, fully dense retriever under a limited computational resource setup.

Inspired by the advances in pre-trained large language models, recent work in multi-hop retrieval has proposed to formulate the problem in a fully generative way. In particular, Lee et al. [80] proposed an encoder-decoder model that performs multi-hop retrieval by simply generating the entire text sequences of the retrieval targets (e.g., document). In contrast to the fully dense retrieval approaches [175] where the documents and the questions interact in the L2 or inner product space (i.e., the document and the question encoders map the documents and the questions to a common space and perform a nearest neighbor search), in generative multi-hop retrieval, they interact in the language model's parametric space.

Part IV

Improving Domain Adaptation in KGQA with Neural Retrieval

8

Improving Relation Prediction with Synthetic Data and Neural Retrieval

In the last part of this thesis, we study how search engines can utilize knowledge graphs (KGs) in order to answer queries. We set our study in the context of question answering (QA) over KGs. In particular, we focus on a setting where new domains (i.e., subgraphs) covering unseen–and rather different to existing domains–relations and entities are added to the KG. For these new domains, there are no available query-answer pairs during training time.

QA systems over KGs commonly take the form of a pipeline consisting of entity mention detection, entity candidate generation, relation prediction, and answer selection. Preliminary experiments suggested that relation prediction does not scale to new domains out-of-the-box. Therefore, in this chapter, we aim to answer **RQ7**: Can neural retrieval combined with data augmentation increase the robustness of relation prediction in a KGQA system over previously unseen domains?

8.1 Introduction

Large-scale structured knowledge graphs (KGs) such as Freebase [10] and Wikidata [119] store real-world facts in the form of subject–relation–object triples. KGs are being increasingly used in a variety of tasks that aim to improve user experience [12]. One of the most prominent tasks is knowledge graph simple question answering (KGSQA), which aims to answer natural language questions by retrieving KG facts [181]. In practice, many questions can be interpreted by a single fact in the KG. This has motivated the KGSQA task [11, 110, 120], which is the focus of this chapter. In KGSQA, given a *simple* question, e.g., "who directed the godfather?", the system should interpret the question and arrive at a single KG fact that answers it: (The Godfather (film), film.film.directed_by, Francis Ford Coppola).

KGSQA systems are trained on manually annotated datasets that consist of question-fact pairs. In practice, the applicability of such systems in the real-world is limited by two factors: (i) modern KGs store millions of facts that cover thousands of different

This chapter was published as G. Sidiropoulos, N. Voskarides, and E. Kanoulas. Knowledge graph simple question answering for unseen domains. In *Conference on Automated Knowledge Base Construction, AKBC*, 2020.

relations, but KGSQA training datasets can only cover a small subset of the existing relations in the KG [37], and (ii) KGs are dynamic, i.e., they are updated with new domains that cover new relations [119]. Solving (i) and (ii) by exhaustively gathering question-fact pair annotations would be prohibitively laborious, therefore we need to rely on automatic methods.

Motivated by the above, in this chapter, we study the KGSQA task in a setting where we are interested in answering questions about a new, unseen domain that covers relations, for which we *have* instances in the KG, but we *have not seen* any question-fact pair during training. We model this as a domain adaptation task [98, 117] and propose a data-centric domain adaptation framework to address it. Data-centric domain adaptation approaches focus on transforming or augmenting the training data, instead of designing specialized architectures and training objectives as model-centric domain adaptation approaches do [20]. Our framework consists of: (a) a KGSQA system that can handle the unseen domain, and (b) a novel method that generates training data for the unseen test domain.

The KGSQA system we introduce performs mention detection, entity candidate generation, and relation prediction on the question, and finally selects the fact that answers the question from the KG. Our preliminary experiments showcased that even though mention detection, entity candidate generation, and answer selection generalize well on new domains, relation prediction does not. To improve the performance of relation prediction on questions that cover relations of the unseen domain, we (i) treat relation prediction as a retrieval task and employ a neural retriever model and (ii) automatically generate synthetic questions from KG facts (i.e., knowledge graph question generation – QG) of the unseen domain to use for training. The resulting synthetic question-fact pairs are used to train the KGSQA system for the unseen domain. We find that the effectiveness of QG for KGSQA can be restricted not only by the quality of the generated questions, but also by the lexical variety of the questions. This is because users ask questions underlying the same relation using different lexicalizations (e.g. "who is the author of X", "who wrote X"). To address this, we use distant supervision to extract a set of keywords for each relation of the unseen domain and incorporate those in the question generation method.

We break down **RQ7** into three research sub-questions. In particular, we aim to answer:

RQ7.1 How does our method for generating synthetic training data for the unseen domain perform on RP compared to a set of baseline methods?

RQ7.2 How does our full method perform on KGSQA for unseen domains compared to state-of-the-art zero-shot data-centric methods?

RQ7.3 How does our data-centric domain adaptation method compare to a state-of-the-art model-centric method on RP?

Our main contributions are the following: (i) we introduce a new setting for the KGSQA task, over new, previously unseen domains, (ii) we propose a data-centric domain adaptation framework for KGSQA that is applicable to unseen domains, and (iii) we use distant supervision to extract a set of keywords that express each relation

of the unseen domain and incorporate them in QG to generate questions with a larger variety of relation lexicalizations. We experimentally evaluate our proposed method on a large-scale KGSQA dataset that we adjust for this task and show that our proposed method consistently improves performance over zero-shot baselines and is robust across domains.[1]

8.2 Problem Statement

Let E denote the set of entities and R the set of relations. A KG K is a set of facts (e_s, r, e_o), where $e_s, e_o \in E$ are the subject and object entities respectively, and $r \in R$ is the relation between them. Each relation r has a unique textual label r_l and falls under a single domain $\mathcal{D}$. For instance, music.album.release_type and music.artist.genre fall under the Music domain. *Simple* questions mention a single entity and express a single relation. For instance, the question "who directed the godfather?" mentions the entity "The Godfather" and expresses the relation film.film.directed_by. Given a *simple* question q that consists of a sequence of tokens $t_1, t_2, \ldots, t_T$, the KGSQA task is to retrieve a fact $(\hat{e}_s, \hat{r}, \hat{e}_o)$, where $(\hat{e}_s, \hat{r})$ accurately interprets q (i.e., $\hat{e}_s$ is mentioned in q and $\hat{r}$ is expressed in q) while $\hat{e}_o$ provides the answer to q.

In our setting, we aim to build a KGSQA system that can perform well on a previously unseen domain. A domain is "unseen" when facts that cover relations of that domain do exist in K, but gold-standard question-fact pairs of that domain do *not* appear in the training data. This setting is an instance of domain adaptation, where a model is trained on data $\mathcal{S}$, drawn according to a source distribution, and tested on data $\mathcal{T}$ coming from a different target distribution. Domain adaptation over KG domains is more challenging compared to domain adaptation over single KG relations [171, 184], because it is less likely for relations with similar lexicalizations to appear in the training set.

8.3 KGSQA System

In this section, we detail our KGSQA system. In Section 8.4, we will describe how we generate synthetic training data to make this system applicable to unseen domains. Following the current state-of-the-art on KGSQA [120], we split the task into four sub-tasks, namely, *entity mention detection* (MD), *entity candidate generation* (CG), *relation prediction* (RP), and *answer selection* (AS). The skeleton of our KGSQA system generally follows previous work, and we modify the MD and RP architectures.

Mention Detection (MD)

Given the question q, MD outputs a single entity mention m in q, where m is a subsequence of tokens in q. We model this problem as sequence tagging, where given a sequence of tokens, the task is to assign an output class for each token [67, 79]. In our

[1] Our code is available at `https://github.com/GSidiropoulos/kgsqa_for_unseen_domains`.

case, the output classes are entity (E) and context (C). For instance, the correct output for the question "who directed the godfather?" is "[C C E E]". We use a BiLSTM with residual connections (R-BiLSTM) [58], since it outperformed vanilla RNN, BiRNN, and a CRF on top of a BiRNN [120] in preliminary experiments.

Candidate Generation (CG)

Given the mention m extracted from the previous step, CG maps m to a set of candidate entities $C_S \subset E$. For instance, CG maps the mention "the godfather" to the entities $\{$ The Godfather(film), The Godfather(book)$\dots\}$. The CG method we use was proposed in Türe and Jojic [157].

Briefly, the method pre-builds an inverted index I from n-grams of mentions to entities, and it looks-up the n-grams of m in I to obtain C_S. We pre-built an inverted index, I, that maps all n-grams ($n \in \{1, 2, 3, \dots, \infty\}$) of an entity name to the entities that share the specific name (partially or not), each accompanied by a tf-idf score. Tf-idf is used to determine the importance of the n-gram to the entity name; the latter acts as the document while the former as the term. We then produce all the corresponding n-grams of the mention m obtained from ED, and then search for them in I. Starting from the highest order n-gram we retrieve entities and append them to the entity candidate set C_S, favoring those with the highest score. If we find an exact match for an entity, we do not further consider lower-order n-grams, backing off otherwise. Additionally, and different from [157], we stop searching for entities if the unique entity names within C_S are equal to the length of the mention.

Relation Prediction (RP)

Given the question q and the set of entities C_s extracted in the previous step, RP outputs a single relation $\hat{r} \in R$ that is expressed in q. Previous work models RP as a large-scale multi-label classification task where the set of output classes is fixed [120]. In our domain adaptation scenario, however, we want to be able to predict relations that we have not seen during training. Therefore, we model RP as a relation retrieval task, as in [184], and use the textual label r_l to represent the relation r (instead of using a categorical variable). This way we can in principle represent any relation $r \in R$ during inference time. Our architecture is a simpler version of the one presented in the work by Yu et al. [184], in which the authors formulate a relation both as a sequence and a categorical variable and use more complex sequence encoders. Below we describe the architecture we use for RP and how we perform training and inference.

We follow a neural retrieval approach with a dual-encoder architecture consisting of a question encoder and a relation encoder. The neural encoder $Enc_Q(\cdot)$ maps any question to a low-dimensional real-valued vector, and similarly, $Enc_R(\cdot)$ maps any relation to a low-dimensional real-valued vector. The similarity between a question and a relation is computed using a similarity metric on the respective vectors.

Encoding To increase the model's generalization ability beyond specific entity names, we first replace the previously detected entity mention m in q with a placeholder token, for example "who directed SBJ". To represent a relation r, we use its label r_l

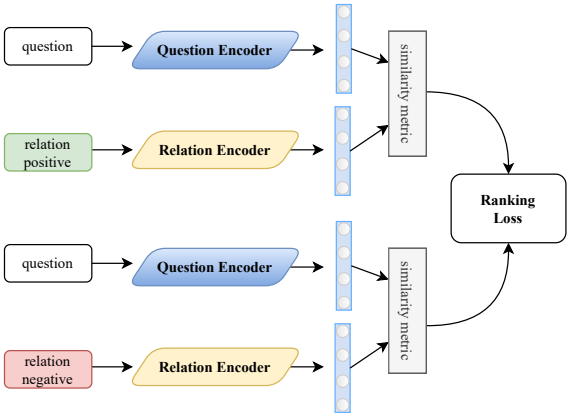


Figure 8.1: Training overview of relation prediction.

(e.g., film.film.directed_by). The question and the relation neural encoders encode the question q and the relation r:

$$\boldsymbol{\gamma}^{(q)} = Enc_Q(q), \tag{8.1}$$

$$\boldsymbol{\gamma}^{(r)} = Enc_R(r_l). \tag{8.2}$$

Since questions and relations significantly differ grammatically and syntactically, the two encoders do not share any parameters. In particular, we use two independent LSTM-based encoders.

At this point, we provide additional information on how the question and relation representations are obtained. Each question term is mapped to a word embedding, and subsequently, the word embeddings are fed to an LSTM; embeddings are initialized with pre-trained word2vec embeddings [104]. The final hidden state of the LSTM $\boldsymbol{\gamma}^{(q)}$ is used as the representation of the question. Similarly, for relation, we encode r_l with an LSTM to obtain $\boldsymbol{\gamma}^{(r)}$.

Training We train our retriever using standard pairwise learning to rank. The ranking function f is calculated as $f(q, r) = \cos(\boldsymbol{\gamma}^{(q)}, \boldsymbol{\gamma}^{(r)})$, where $\cos(\cdot)$ is the cosine similarity. The loss is defined as follows:

$$L(\theta) = \sum_{r} \sum_{r' \in R'} \max(0, \mu - f(q, r) + f(q, r')), \tag{8.3}$$

where θ are the parameters of the model, μ is a hyperparameter, and R' is the set of sampled negative relations for a question q. Figure 8.1, visualizes the training process.

Sampling negatives We design a specialized *negative sampling* method to select R'. With probability P_R^- we uniformly draw a sample from $R^- = \{r' | r' \in R \wedge r' \neq r\}$,

the set of all available relations except the positive relation r. With probability $1 - P_R^-$ we draw a random sample from $\hat{R}^- = \{r' | r' \in \mathcal{D}_R^+ \wedge r' \neq r\}$, the set of relations that are in the same domain as the positive relation r. This way, we expose the model to conditions it will encounter during inference.

Inference At inference time, given a question q and a set of relations, we score all question-relation pairs (q, r) with f and select the relation $\hat{r}$ with the highest score. Unfortunately, computing a score with respect to all possible relations in R leads to poor performance when there is no linguistic signal to disambiguate the choice. In order to address this issue, we constrain the set of the potential output relations R_c to be the union of the relations expressed in the facts in which the entities in C_S participate [120]. Formally, we define the target relation classes to be $R_c = \{r \in R | (e_s, r, e_o) \in K \wedge e_s \in C_S\}$. For example, given the question "who directed the godfather", the potential relations are { film.film.directed_by, book.written_work.author, ... }. Using the aforementioned constraint we can safely ignore relations like tv.tv_series_episode.director by taking into account that The Godfather does not appear in any tv-related facts.

Answer Selection (AS)

Given the set of entities C_S obtained from CG, and the top-ranked relation $\hat{r}$ obtained from RP, AS selects a single fact $(\hat{e}_s, \hat{r}, \hat{e}_o)$, where $\hat{e}_o$ answers the question q. The set of candidate answers may contain more than one fact $(e'_s, \hat{r}, e'_o)$, where $\forall e'_s \in C_S$. Since there is no explicit signal on which we can rely to disambiguate the choice of subject, all the potential answers are equally probable. Therefore, we use a heuristic based on popularity, introduced by Mohammed et al. [110]: we choose $\hat{e}_s$ to be the entity that appears the most in the facts in K either as a subject or as an object. Having $\hat{e}_s$ and $\hat{r}$ we can retrieve the fact $(\hat{e}_s, \hat{r}, \hat{e}_o)$. For our running example ("who directed the godfather"), given film.film.directed_by (from RP) and entities { The Godfather(film), The Godfather(book)... } (from CG) we can select the fact (The Godfather (film), film.film.directed_by, Francis Ford Coppola).

8.4 KGSQA to Unseen Domains Using Question Generation

Even though all the components of the aforementioned KGSQA system were designed to work with unseen domains, preliminary experiments demonstrated that RP does not generalize well to questions originating from unseen domains. This is expected since RP is a large-scale problem with thousands of relations, making it challenging to model less frequent or even unknown relations that are expressed with new lexicalizations [37].

We therefore focus on improving RP for questions originating from unseen domains. Inspired by the recent success of data-centric domain adaptation in neural machine translation [20], we perform synthetic question generation from KG facts of the unseen

domain to generate question-fact pairs for training the RP component (see Section 8.3).[2] In the remainder of this section, we briefly describe the base question generation (QG) model we build upon and how we augment the model to more effectively use textual evidence and thus better generalize to relations of the unseen domain.

8.4.1 Base model for question generation

Given a fact (e_s, r, e_o) from the target domain, QG aims to generate a synthetic question $\hat{q}$. During training, only question-fact pairs from the known domains are used. Our base model is the state-of-the-art encoder-decoder architecture for QG [37]. It takes as input the fact (e_s, r, e_o) alongside a set of textual contexts $C = \{c_s, c_r, c_o\}$ on the fact. Those textual contexts are obtained as follows: c_s and c_o are the types of entities e_s and e_o respectively, whereas c_r is a lexicalization of the relation r obtained by simple pattern mining on Wikipedia sentences that contain instances of r. For instance, given the fact (The Queen Is Dead, music.album.genre, Alternative Rock), the textual contexts are: $c_s = \{$"album"$\}$, $c_r = \{$"album by"$\}$ and $c_o = \{$"genre"$\}$.

The encoder maps e_s, r and e_o to randomly initialized embeddings and concatenates those to encode the whole fact. Also, it encodes the text in c_s, c_r and c_o separately using RNN encoders. The decoder is a separate RNN that takes the representation of the fact and the RNN hidden states of the textual contexts to generate the output question $\hat{q}$. It relies on two attention modules: one over the encoded fact and one over the encoded textual contexts. The decoder generates tokens not only from the output vocabulary but also from the input (using a copy mechanism) to deal with unseen input tokens.

8.4.2 Using richer textual contexts for question generation

The role of the textual contexts C in the aforementioned base model is critical since it enables the model to provide new words/phrases that would have been unknown to the model otherwise [37]. Even though the base model generally generates high-quality questions, in our task (KGSQA), we aim to generate a larger range of lexicalizations for a single relation during training in order to generalize better at test time. This is because users with the same intent may phrase their questions using different lexicalizations (e.g., "who is the author of X", "who wrote X"). Thus, in this section, we focus on how to provide the model with a diverse set of lexicalizations for a relation r instead of a single one as in the base model, in order to be able to generate a more diverse set of questions in terms of relation lexicalizations. More precisely, given a relation r, we extract k keywords that will constitute the relation's textual context c_r. To this end, we first extract a set of candidate sentences S_r that express a specific relation r between different pairs of entities. Second, we extract keywords from the set S_r, rank them, and select the top-k keywords that constitute the set c_r. We detail each of these steps below.

Extracting sentences Given a set of facts F_r of relation r between different pairs of entities, we aim to extract a set of sentences S_r, where each sentence $s \in S_r$ expresses

[2] Note that Dong et al. [36] also performed QG for improving the overall KGSQA performance. However, their model is not applicable to our domain adaptation scenario since it relies on modifying existing questions and all domains were predefined.

Table 8.1: Examples of textual contexts extracted by our keyword extraction approach.

Relation	Textual Context
music.artist.label	records, artists, album, released, label, signed, band
film.film.directed_by	film, director, directed, films, short, directing, producer
people.deceased_person.place_of_death	died, death, deaths, born, age, people, male, actors

a single fact (e_s, r, e_o) in F_r [162]. For this, for each fact (e_s, r, e_o) in F_r, we need to (a) extract a set of candidate sentences S that might express (e_s, r, e_o) and (b) select the sentence that best expresses the relation. For (a), we collect the set of sentences S using distant supervision, similarly to [106]: S consists of sentences that mention e_o in the Wikipedia article of e_s and sentences that mention e_s in the Wikipedia article of e_o. For (b), we score each sentence $s \in S$ w.r.t. the label r_l of the relation r using the cosine similarity $cos(e(s), e(r_l))$, where $cos(\cdot)$ is the cosine similarity and $e(x)$ is calculated as $e(x) = (1/|x|) \sum_{t \in x} w_t$, where w_t is the embedding of word t. Finally, we take the sentence s' with the highest score and add it to the set S_r.

Extracting keywords After extracting the set of sentences S_r, we aim to extract the set of keywords c_r. For this, we treat S_r as a single document and score each word t that appears in S_r using tf-idf, $\text{score}(t) = \text{tf}(t, S_r) \cdot \text{idf}(t, S_R)$, where S_R is the union of all $S_{r'}, r' \in R$. The top-k scoring words constitute the set of keywords c_r. Table 8.1 depicts example keywords generated by the procedure described above.

The keyword extraction approach described above is conceptually simple yet we later show that it significantly improves upon the base model when applied to KGSQA.

8.5 Experimental Setup

In this section, we discuss how we design the experiments to answer our research questions.

8.5.1 Dataset and metrics

Dataset In our experiments we use the SimpleQuestions dataset, which is an established benchmark for studying KGSQA [11]. The dataset consists of 108,442 questions written in natural language by human annotators, paired with the ground truth fact that answers the question. The ground truth facts originate from Freebase [10]. The dataset covers 89,066 unique entities, 1,837 unique relations and 82 unique domains. In our setup, we leave one domain out to simulate a new, previously unseen domain, and train on the rest. We choose six challenging domains as target domains: Film, Book, Location, Astronomy, Education and Fictional Universe; the first three are among the largest domains and the last three are medium-sized. The aforementioned domains are challenging because they have very low overlap in terms of relation lexicalization w.r.t. the rest of the domains used as source domains. The training data consists of the

question-fact pairs that appear in the source domains, augmented with synthetically generated data of the target/unseen domain. In practice, we replace all questions from the target domain that initially appear in the full training set with their corresponding synthetically generated questions.[3] As a source of text documents for the textual context collection (see Section 8.4.2), we use Wikipedia articles augmented with dense entity links provided by DAWT [147].

Evaluation metrics We run the experiments three times and report the median (only marginal and not significant differences were found among different runs) [110]. In contrast to the classic KGSQA where the task is to retrieve a single entity, it is standard practice when using the SimpleQuestions dataset to treat the problem as question interpretation [120]. More specifically, the objective is to rewrite the natural language question in the form of a subject-relation pair. We evaluate our overall approach in terms of top-1 accuracy, i.e., whether the retrieved subject-relation pair matches the ground truth. We measure accuracy both at a macro- (domains) and at a micro-level (samples). Statistical significance is determined using a paired two-sided t-test.

8.5.2 Baselines

Question generation baselines To answer **RQ7.1**, we examine the impact–in RP performance–of different methods for generating synthetic questions. To test the different methods in a fair setting, we keep the KGSQA system unchanged and alter the way of generating synthetic questions. We compare the RP performance on the unseen domain given the following ways of generating synthetic data of the unseen domain:

- No synthetic data: there are no questions for the unseen domain.
- Wiki-raw-sentences: uses the raw Wikipedia sentence that expresses the ground truth fact that answers the question (automatically extracted using the procedure in Section 8.4.2).
- QG: the state-of-the-art QG method proposed by Elsahar et al. [37] (see Section 8.4.1).

Relation prediction baselines To answer **RQ7.2**, we replace our RP component with two state-of-the-art RP models:

- BiLSTM: follows a traditional approach for RP, treating the problem as a classification task and employing a BiLSTM to classify relations [120].
- HR-BiLSTM: treats RP as a retrieval problem and follows a neural retriever approach where the encoder is a hierarchical residual BiLSTM (HR-BiLSTM) [184]; furthermore, it is specifically designed to deal with unseen or less frequently seen relations.

[3] One may hypothesize that since entities can appear in multiple domains (e.g. actors who are also singers), question generation becomes an unrealistically simple task. However, this is not the case because in our dataset, the entity overlap between seen and unseen domains is *only* 4.6%.

Model-centric baseline To answer **RQ7.3**, we compare the performance of our data-centric model on RP against a state-of-the-art model-centric zero-shot approach [171]: the HR-BiLSTM proposed by Yu et al. [184] with an adversarial adapter combined and a reconstruction loss. The adapter uses embeddings trained on Freebase and Wikipedia by JointNRE [54] and learns representations for both seen and unseen relations.

8.5.3 Implementation details

As mentioned in Section 8.3, we employ an R-BiLSTM for MD. Our MD model consists of 2 hidden layers, 600 hidden units, 0.4 dropout rate, frozen embeddings, and a learning rate of 10^{-3}. We train for 50 epochs. For RP we use an LSTM-based neural retriever (see Section 8.3). In detail, we use a 1-layer LSTM for the question and the relation encoders; consisting of 400 hidden units, a frozen embedding layer, and a learning rate of 10^{-3}. We train for 10 epochs. During training, we sample 10 negative questions per question using the procedure described in Section 8.3. We initialize word embeddings with pre-trained Google News 300-dimensional embeddings [104]. We use the Adam optimizer [76]. We use a batch size of 300 and 200 for MD and RP, respectively.

For the QG model [37] and the model-centric RP [171] model we compare against, we use the hyperparameters as presented in their work. Note that for both our method and the baselines, the hyperparameters were tuned on the validation split of the SimpleQuestions dataset. We keep the parameters fixed for both our method and the baselines for all source-target domain setups. We set the number of keywords for each relation $k = 10$ (Section 8.4.2).

8.6 Results and Discussion

In this section we present and discuss our experimental results. All models under comparison have all their components fixed, except RP. Therefore, any improvement observed is due to RP.

Effect of synthetic data on RP (RQ7.1) Here we compare the RP performance of our method for generating synthetic training data with a set of baselines. For this experiment, the RP component of the KGSQA system remains unchanged and we only alter the data it is trained with. Table 8.2 shows the results. We observe that our QG method is the best-performing one. It significantly outperforms the baseline QG method, which confirms that our method for generating rich textual contexts for relations (Section 8.4.2) is beneficial for KGSQA. As expected, using the raw Wikipedia sentence that expresses the ground truth fact that answers the question as synthetic questions (i.e., Wiki-raw-sentences) performs better than when not using training data from the target domain at all but performs much worse than the QG methods. This is expected since Wikipedia sentences are very different both syntactically and grammatically from the real questions that the KGSQA system encounters during test time. Next, we investigate how RP performance varies depending on the number of the target domain questions used to augment the training set. Figure 8.2 shows the results. First, we observe that, in the low data regime (less than 100 questions), the gap in performance between training with

Table 8.2: Relation Prediction accuracy w.r.t. different ways of generating synthetic training data for the unseen domain. ▲ indicates a significant increase in performance compared to the top performing baseline ($p < 0.01$).

Synthetic training data	Macro-avg. Accuracy (%)	Micro-avg. Accuracy (%)
-	30.21	29.06
Wiki-raw-sentences	37.89	36.51
QG [37]	67.52	69.78
QG (Ours)	**69.86**▲	**70.95** ▲

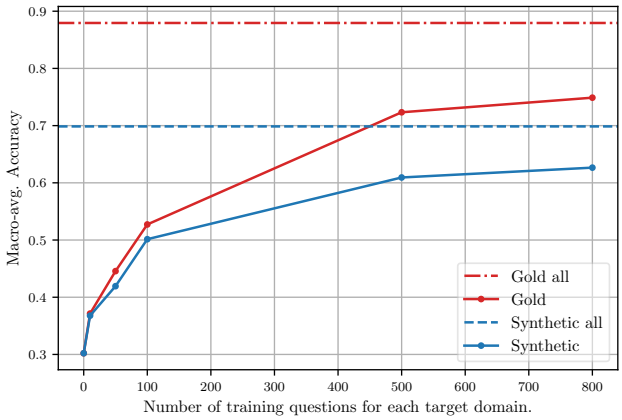


Figure 8.2: RP macro-accuracy when varying the number of target domain questions used to augment the training set. Gold refers to the gold standard questions and synthetic refers to the automatically generated questions. Gold all (Synthetic all) refers to the full set of training gold (synthetic) questions. The training set size of the smallest domain is 800, thus we report performance up to that point.

gold or synthetic questions is small. This is encouraging for applying our framework on domains in the long tail. As the number of questions increases, the performance for both gold and synthetic increase, however the gap between them increases, which is expected.

Overall KGSQA performance for data-centric methods (RQ7.2) Next, we compare our full framework to variations that use state-of-the-art RP models. Table 8.3 shows the results. We observe that our full method (second to last row) improves over all the baselines and significantly outperforms the best-performing baseline. As expected, we see that even though our full method has strong generalization ability for unseen domains, there is a gap in performance when using the automatically generated synthetic questions (second to last row) or the human-generated questions (last row). This gap suggests that QG has room for improvement.

Next, we test the systems under comparison in terms of generalization ability across

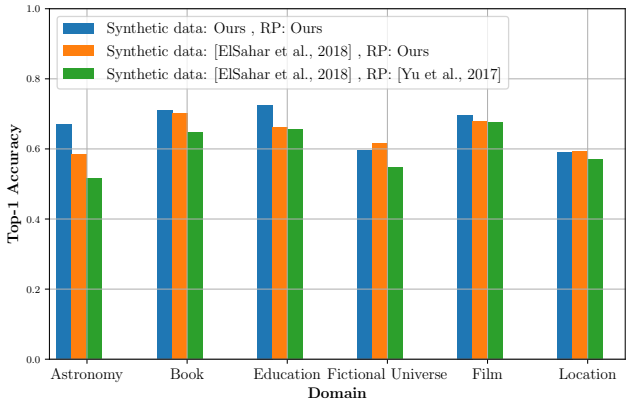


Figure 8.3: End-to-end accuracy on the KGSQA task per domain.

Table 8.3: End-to-end accuracy on the KGSQA task. ▲ indicates a significant increase in performance compared to the top-performing baseline ($p < 0.01$).

Synthetic training data	Relation Prediction (RP)	Macro-avg. Accuracy (%)	Micro-avg. Accuracy (%)
QG [37]	BiLSTM [120]	55.49	55.11
	HR-BiLSTM [184]	60.20	62.77
	Ours	63.90	65.18
QG (Ours)	Ours	**66.49**▲	**66.64**▲
Gold Questions	Ours	84.56	82.87

domains. Figure 8.3 shows the results. First, we observe that our method achieves an accuracy of at least 60% for all domains, which shows that it is robust across domains. Also, it outperforms the baselines in all but one domain. In order to gain further insights, we sampled success and failure cases from the test set. We found that the errors in the failure cases generally originate from the fact that the model relies on lexicalizations that are frequent in the seen domains. We show such cases in Table 8.4. Furthermore, our analysis showed that one way of improving QG is to improve keyword extraction by collecting a larger set of relevant sentences that express a single relation, possibly by looking into other sources of text (e.g., news articles). In Table 8.5, the automatic evaluation results for the synthetic questions generated by our QG model against those generated by [37] further strengthen our claim. As can be seen from the table, our model outperforms the baseline, indicating that using a larger set of relevant sentences for a single relation, can be beneficial to the generated questions.

Table 8.4: Examples of success cases (top 3 rows) and failure cases (bottom 3 rows) of our QG method.

Unseen Domain	Gold Questions	Synthetic Questions
Astronomy	what is something that carolyn shoemaker discovered	what is the astronomical objects discovered by carolyn shoemaker
Book	what's the subject of the cognitive brain	what is the subjects of the written work the cognitive brain
Location	which country is cumberland lake located in	where is cumberland lake located
Film	in what country did the film joy division take place	what country is joy division under
Book	who authored the book honor thyself	who was the director of the book honor thyself
Location	what was a historic atlantic city convention hall team	what event took place at historic atlantic city convention hall in 1943

Table 8.5: Automatic evaluation of question generation w.r.t. BLEU.

	BLEU-1	BLEU-2	BLEU-3	BLEU-4
QG [37]	44.04	28.63	16.50	9.13
QG (Ours)	**44.27**	**29.55**	**17.73**	**10.16**

Comparison to a model-centric method (RQ7.3) Here, we compare our data-centric method for domain adaptation to a state-of-the-art model-centric method on RP [171]. In order to perform a fair comparison when testing for RP, we follow their setup (see Section 5.1 in [171]) and for this particular experiment, we assume that MD and CG produce the correct output. Our method outperforms their method both on macro-accuracy (75.54% vs. 75.02%), and micro-accuracy (77.08% vs. 72.17%). Note that we use randomly initialized embeddings whereas in the work by Wu et al. [171] the authors use JointNRE relation embeddings trained on Wikipedia and Freebase, which provides an advantage to their method. Also, note that their method (model-centric) is orthogonal to ours (data-centric), and therefore, an interesting future work direction would be to explore how to combine the two methods to improve performance further.

Qualitative error analysis In order to gain insights on how each part of the pipeline affects the final prediction, we perform an empirical error analysis. We sample 60 examples for which our system provided a wrong answer (10 for each target domain) and investigate what led to the wrong prediction. Out of these examples, 43 mistakes were due to RP, 6 due to ED, 5 due to CG, and 6 due to AS. For RP, 22 mistakes were assigned to conceptually similar relations within the target domain, 15 to similar relations outside the target domain, and the rest to a common relation outside the target domain. For ED, 4 mistakes were due to predicting an extra token as part of the entity mention and 2 were due to missing a token from the entity mention. For CG, 2 mistakes were mistakes because the gold entity was not part of the mention candidates and 1 was due to an error in the human annotation; the early termination proposed in [157], is responsible for the remaining mistakes. This analysis confirms that RP remains the most challenging part of KGSQA. Within RP, the challenge seems to be that there are relations for which there is a high lexical similarity between the corresponding questions and also between the relations per se.

KGSQA performance on seen domains Finally, even though the focus of this chapter is to perform KGSQA on unseen domains and thus we do not aim to improve state-of-the-art on seen domains, we also test our KGSQA system on the standard split of the SimpleQuestions dataset. Our model achieves a top-1 accuracy of 77.0%, which is ranked third among the state-of-the-art methods while having a simpler method than the two top-performing ones [51, 120]. We provide a thorough comparison w.r.t. the state-of-the-art on seen domains in Table 8.6.[4]

[4]Note that Zhao et al. [192] reported an accuracy of 85.44%. However, they calculate accuracy w.r.t. the correctness of the object entity, which is not standard when testing on the SimpleQuestions dataset (see Section 8.5.1). When we calculate accuracy that way, Petrochuk and Zettlemoyer [120] achieves an accuracy of 91.50% and our method achieves 87.31%.

Table 8.6: Top-1 KGSQA accuracy on seen domains.

Model	Accuracy (%)
Random guess [11]	4.9
Memory NN [11]	62.7
Attn. LSTM [59]	70.9
GRU [95]	71.2
BiGRU-CRF & BiGRU [110]	74.9
CNN & Attn. CNN & BiLSTM-CRF [182]	76.4
HR-BiLSTM & CNN & BiLSTM-CRF [184]	77.0
Ours	**77.0**
BiLSTM-CRF & BiLSTM [120]	78.1
Solr & TSHCNN [51]	80.0

8.7 Related Work

Methods on the standard KGSQA task are split into those following a pipeline approach–MD, CG, RP & AS [110, 120, 157] or an end-to-end approach [51, 95]. In our work, we follow the former approach for solving KGSQA on unseen domains, since we found that all the components except RP are relatively robust for unseen domains. We leave the exploration of end-to-end approaches for our task for future work. More related to our setting, Yu et al. [184] and Wu et al. [171] tackle RP for KGSQA on unseen relations (instead of whole domains). Both are model-centric domain adaptation approaches, while ours is data-centric. We experimentally showed that we outperformed both in the KGSQA setting on unseen domains. Combining model-centric and data-centric approaches for our task would be an interesting future work direction.

More broadly, our work is also related to cross-domain semantic parsing [60, 149, 185, 190]. In contrast to the aforementioned line of work that maps questions to executable logical forms, we focus on questions that can be answered with a single KG fact.

8.8 Conclusion

In this chapter, we proposed a data-centric domain adaptation framework for KGSQA that is applicable to unseen domains. We put most of our efforts into robustifying the RP component for questions originating from the unseen domain with the least generalization ability. To this extent, we proposed an RP method that combines (i) neural retrieval with (ii) data augmentation–to generate synthetic training data for the unseen domains. Concerning the generation of synthetic questions, we further proposed a keyword extraction method that, when integrated into our QG model, allows it to generate questions with various lexicalizations for the same underlying relation, thus better resembling the variety of real user questions. Our experimental results on the SimpleQuestions dataset show that our proposed framework significantly outperforms state-of-the-art zero-shot baselines, and is robust across different domains.

For our data-centric RP approach, we mainly focused on generating better synthetic questions. Although our simple neural retriever for RP can perform well in new domains, our method would greatly benefit from using a more advanced model. Therefore, a potential future direction is to explore a combination of model-centric and data-centric approaches for robust RP. On the other end of the spectrum, the variety of lexicalizations in the synthetic questions is essential for training a model that can be more robust on questions from the new domains. We found that there is room for further improving QG, particularly for KGSQA, which is a promising direction for future work.

In this chapter, we focused on improving domain adaptation in KGQA—with neural retrieval and data augmentation—to assist search engines in accessing structured knowledge. This was the final research chapter of the thesis. In the next chapter, we will summarize our main findings and formulate directions for future work.

8.9 Reflections

Since the research we presented in this chapter was conducted, major changes in the fields of NLP and IR have directly or indirectly impacted our work. In this section, we discuss advances regarding the sub-tasks of relation prediction and question generation, as well as the task of zero-shot KGQA.

In this chapter, we followed an LSTM-based approach for relation prediction. However, nowadays, with the advances in pre-trained large language models (e.g., BERT [32]) and training methodologies (e.g., batch contrastive learning and knowledge distillation), there is a plethora of effective first-stage neural retrieval and neural re-ranking models [64, 73, 74]. These models could be used for relation prediction and (potentially) lead to better results.

Similarly, the base model we use for question generation is an RNN-based encoder-decoder model with attention and delexicalization [37]. In recent years, pre-trained language models have increasingly been successfully applied to generate questions from KGs [49, 53, 173]. For instance, Han et al. [53] adapted BART [84] for question generation from RDF input and achieved higher-quality synthetic questions (in terms of BLEU score). Furthermore, recent works on question generation have explicitly focused on improving the lexicalization variety in the generated questions [50].

Finally, over the past years, KGQA frameworks designed explicitly to generalize to unseen domains have been proposed. An example is the work by Mckenna and Sen [102]. The authors proposed an end-to-end KGQA model–which follows an encoder-decoder architecture and uses differentiable KG [26]–that can handle new relations and entities at test time without retraining.

Appendix 8.A

In this chapter, we used deep neural networks, such as LSTMs, as the backbone of our models. However, we can use more advanced models such as BERT. We tried to investigate if using BERT could result in performance gains. To do so, we ran a preliminary experiment where we replaced the relation prediction (RP) module–the

Table 8.7: Comparison of relation prediction accuracy using our LSTM-based neural retriever vs. a BERT-based ranker.

Setting	Model	Macro-avg. Accuracy (%)	Micro-avg. Accuracy (%)
zero-shot	Ours	30.21	29.06
	BERT	47.62	45.29
synthetic	Ours	69.86	70.95
	BERT	59.08	59.57
gold	Ours	87.85	88.76
	BERT	81.92	84.90

most important component of the pipeline–with a BERT-based ranker. The input to BERT is the question and the relation separated by [SEP] (e.g. [CLS] who wrote pulp fiction [SEP] film film written by [SEP]), and we predict a binary label using a linear layer on top of the [CLS] token.

We explored what we could achieve with a BERT architecture in three setups: (i) adding no new training data for the target domain (zero-shot), (ii) using our QG method to generate training data for the target domain (synthetic), and (ii) using gold standard training data from the target domain (gold). We report the results in Table 8.7.

We found that BERT performs best on new, unseen data when no gold or synthetic data is provided, showing a $\sim$17% improvement over our model. We also observe that when synthetic or gold data is provided the BERT performance improves compared to the zero-shot scenario. Interestingly, its performance is worse than our model's (LSTM-based) performance when synthetic or gold data are used. We hypothesize that this is because BERT needs a larger amount of data for fine-tuning. On the other hand, our model does not generalize as well without gold or synthetic data, but it can quickly adapt to new domains when provided with such data. It is worth noting that adding synthetic data from the target domain has a positive impact on both models (BERT and ours).

9 Conclusions

In this thesis, we focused on developing robust neural retrievers. The study is structured around four main research themes, namely: (i) improving the robustness of neural retrievers against typos, (ii) improving the robustness of speech-based search with neural retrieval, (iii) improving the efficiency of training neural retrievers on low resources for multi-hop retrieval, and (iv) improving domain adaptation in KGQA with neural retrieval. In this chapter, we summarize our main findings concerning the research questions from Chapter 1 and suggest future directions.

9.1 Main Findings

9.1.1 Improving the robustness of neural retrievers against typos

RQ1 Can we robustify dense retrievers against queries with typos?

To answer this question, we proposed an alternative training setup for the dense retriever, which aims to learn better representations from noisy text. Our approach combined data augmentation with an additional robustifying subtask (alongside the main retrieval task) to align the original, typo-free queries with their typoed variants. In more detail, we used synthetically generated queries with realistic typos and employed an additional contrastive loss function that forces the representation of the original typo-free query and its typoed variation to be close and far from other unique queries. Our experimental results showed that our approach improves robustness against typos and performs better than separately applying data augmentation or contrastive learning. Further analysis showcased that typos can have different impacts on the retrieval performance depending on the word they appear in, e.g., typos in highly discriminative utterances as the main entity in a query can have a catastrophic effect while a typo in a stopword may only have a small effect. Moreover, we demonstrated that our model does not simply learn to ignore the words that contain typos but instead learns useful representations from noisy text. However, we showed that learning such representations requires frequently encountering the particular noisy text during the training phase.

During training, the typoed variations of queries are obtained from a realistic typo generator. Therefore, it is possible to have more than one typoed variation per query, which can be used as positive samples. However, most existing methods, including

ours, rely on a single positive sample and a set of negatives per anchor. To this end, they employ classic contrastive learning for the robustifying subtasks, which accounts for single a positive and multiple negatives, thereby limiting the usage of the multiple positives. This finding gave rise to the following research question:

RQ2 Can we improve the robustness of dense retrievers with contrastive learning in a way that accounts for multiple positives and negatives?

To answer this research question, we revisited recent methods in typo-robust dense retrieval, where the robustifying subtasks were tackled via contrastive learning with a single positive and multiple negatives. In more detail, we suggested to use all the available positives simultaneously by adopting a multi-positive contrastive learning approach. Our experimental results showed that multi-positive contrastive learning can further strengthen robustness over conventional contrastive learning with a single positive. Moreover, we found that our multi-positive variant consistently outperformed the original model for the different numbers of typoed variants per query.

Besides learning better representations for noisy text, robustness can originate from uncertainty-aware retrieval models. The recent MRL framework proposed by Zamani and Bendersky [187] was the first method to work toward modeling uncertainty in dense retrieval. It represents queries and documents as multivariate normal distributions rather than vectors and computes query-document similarity as the negative KL divergence between these distributions. This led us to the next research question:

RQ3 Can MRL capture uncertainty in queries that contain typos?

To answer this research question, we had to reproduce the MRL framework since there was no publicly available code or pre-trained model. We unveiled that the MRL framework cannot capture uncertainty in queries with typos. In particular, MRL assigned higher uncertainty to the typo-free queries than typoed ones. Furthermore, even though we showed that MRL can be used to train an effective retriever, most of the findings of the original paper cannot be confirmed. Finally, through our extensive ablation study, we found that the high retrieval performance of MRL does not originate from the multivariate representations but from other components of the framework.

9.1.2 Improving the robustness of speech-based search with neural retrieval

RQ4 Are dense retrievers robust against queries that contain transcription errors?

To answer this research question, we built a large-scale dataset for speech-based search consisting of $\sim 400K$ synthetically generated queries. We assumed an ASR-Retriever pipeline and tested the performance of various classic and state-of-the-art typo-robust dense retrieval models on ASR transcribed queries. Our experimental results showed a dramatic decrease in performance for the classic dense retrievers, while the typo-robust models showed signs of robustness against ASR noise. To further increase the robustness of dense retrievers, we proposed training them on transcriptions from synthetically generated queries, which yielded the best results. Our analysis demonstrated that different accents have different impacts on performance. Finally, we created a new test

set with queries voiced by human users. Using their transcriptions, we demonstrated that the retrieval performance can further degrade when dealing with natural ASR noise instead of synthetic ASR noise.

RQ5 How does a multimodal dense retriever perform in speech-based search?

To answer this research question, we proposed an end-to-end trained, ASR-free, multimodal dense retriever. Our model is constructed based on the classic dense retriever architecture for textual queries and documents, with the introduction of a self-supervised speech model as the query encoder. Our experimental results showed that our retriever is a promising alternative to ASR-Retriever pipelines and competitive on shorter queries. Our analysis revealed the improved robustness of our approach compared to the ASR-Retriever pipeline. We found that the latter's performance is highly dependent on the ASR errors, which can cause significant variations in different scenarios. We observed that pipelines experience a drastic decrease in their retrieval performance in three scenarios: (i) when the ASR model's word error rate increases, (ii) when important words in the spoken question are mistranscribed, and (iii) when there are mistranscriptions that were not encountered during training. Our ASR-free retriever can overcome these issues and provide better results.

9.1.3 Improving the efficiency of training neural retrievers on low resources for multi-hop retrieval

RQ6 Can we train an effective dense retriever for multi-hop retrieval with limited computing resources?

We proposed a hybrid retriever that combines lexical and dense retrieval approaches to answer this research question. In detail, our model consisted of a neural reranker over BM25 retrieved documents for tackling the first hop and a neural retriever for the second hop. Our experimental results showed that end-to-end trained fully dense retrievers experience a significant decrease in performance when trained on limited resources. To this end, we confirmed that our hybrid approach can be trained effectively on limited resources and outperform its fully dense counterparts. In addition, it remained highly competitive with the fully dense retrieval models when the latter were trained on substantially more computational resources.

9.1.4 Improving domain adaptation in KGQA with neural retrieval

RQ7 Can neural retrieval combined with data augmentation increase the relation prediction robustness of a KGQA system over previously unseen domains?

To answer this research question, we formalized the relation prediction task as a retrieval task and used a neural retriever as the underlying model, contrary to most works that treat the problem as classification. This decision allowed for our model to be applicable to previously unseen domains. To further enhance its robustness, we incorporated synthetically generated queries during training to cover relations from the new domains. To generate such queries, we adapted an existing query generation framework by adding

keyword extraction; we used distant supervision to extract a set of keywords that express each relation of the unseen domain and incorporated those into the question generation method. Our experimental results confirmed the effectiveness of our proposed data-centric framework over pure zero-shot baselines. Further analysis showed that our proposed keyword extraction method, integrated into the query generation model, led to generating queries of various lexicalizations for the same underlying relation, thus better resembling the variety of real user queries. Finally, experimentation on the in-domain setting showcased that our entire KGQA system was competitive against state-of-the-art methods.

We now reflect on the main question we asked in Chapter 1, namely, how to improve the robustness and effectiveness of neural retrievers in noisy and low-resource settings. In the first part of this thesis (Chapters 2, 3 and 4), focusing on typoed queries in ad-hoc retrieval, we proposed dense retrieval models that increase robustness against typos by learning better representations for noisy text, and we further examined the typo-robustness of existing uncertainty-aware dense retrieval models. In the second part of this thesis (Chapters 5 and 6), focusing on speech-based search, we examined how dense retrieval models perform–under an ASR-Retriever pipeline scenario–in the presence of ASR transcription errors in the query and proposed a multimodal dense retriever, thereby alleviating some of the limitations of the ASR-Retriever systems. In the third part of this thesis, focusing on answering complex queries under limited computational resources, we proposed a computationally efficient hybrid retriever that combines lexical and dense retrieval and can outperform its fully dense counterparts under limited resources. In the fourth part of this thesis, focusing on relation prediction for KGQA over previously unseen domains, we treated relation prediction as a retrieval task and proposed to use a neural retriever in combination with data augmentation to augment the training set with synthetically generated queries for the new domains, ultimately increasing the overall robustness of the KGQA system on queries originating from the new domains.

9.2 Future Directions

In this section, we discuss the limitations of our study and suggest directions for future work to overcome these limitations and further expand our research.

9.2.1 Improving the robustness of neural retrievers against typos

Typos in entities In Chapters 2 and 3, we introduced dense retrieval models that can be robust against typos by learning better representations for noisy text. Besides their increased performance on queries with typos, we showcased a trend similar to classic dense retrievers (i.e., retrievers not trained explicitly to be robust against typos) on queries with typos on important vs. non-important words, with the models performing significantly better in the latter case. An interesting future direction could involve improving the robustness of dense retrievers when entities are corrupted. Given that dense retrievers have been successfully applied to entity retrieval (in NER) [191], a

potential approach could entail integrating the entity retrieval task while training the typo-robust retriever.

Typos in documents Even though our work (Chapters 2, 3, and 4) focused on the query side, typos can also exist in the documents. Despite the community's focus on improving dense retrievers to handle typos in queries, there are fewer studies on document noise. Therefore, another potential future direction could be to assess the extent to which our proposed approaches are applicable in scenarios involving document noise. Noise appearing in a query can differ from noise in the corpus. For instance, considering the scenario of enabling search in historical documents, optical character recognition (OCR) systems convert physical documents–such as handwritten or printed text–into digital format, allowing users to search these documents. Despite the high quality of state-of-the-art OCR systems, their output is not always correct due to the low resolution of the scanned document, unusual fonts, watermarking, and background noise. As a result, the digitized documents can contain noise. That said, the extent to which methods that address typographical errors in queries can also handle OCR errors in the corpus remains to be seen.

Using LLMs The recent advances in generative large language models (LLMs) have resulted in significant performance improvements in various NLP and IR tasks, in many cases fundamentally transforming the way these tasks are performed. Future research could explore how to use LLMs to build more robust against typos in retrieval systems. This exploration may involve employing zero-shot or few-shot learning with LLMs in several ways, such as (i) to generate more realistic or challenging typos, (ii) to use them as preprocessing steps to clean the query before passing it through the retriever, or even (iii) to create a number of possible corrections for a given typoed query followed by multiple retrieval steps and aggregation of the results.

9.2.2 Improving the robustness of speech-based search with neural retrieval

Multi-vector retrieval In Chapter 6, our multimodal dense retriever demonstrated promising performance compared to its ASR-Retriever counterparts, mainly (i) when the latter faced a high word error rate from the ASR model and (ii) on shorter queries. We speculate that our approach's limited performance on long queries could be attributed to encoding all the information from the speech signal in a single vector. Further investigation into a multi-vector retrieval approach is a potential area for future work.

State-of-the-art ASR The ASR-Retriever pipelines can deliver significantly better results compared to our multimodal method in cases where the ASR model can provide high-quality transcriptions with a low word error rate. Using state-of-the-art ASR models, such as the recently proposed multilingual Whisper [125], can help mitigate ASR errors, particularly in English. However, search is not restricted to the English language and general knowledge queries. Therefore, it is essential to evaluate the effectiveness of scenarios involving low-resource languages, where annotated speech is

scarce, or domain-specific search cases such as medical or legal search. We leave this for future work.

9.2.3 Improving the efficiency of training neural retrievers on low resources for multi-hop retrieval

Embedding bottleneck Typically, dense retrieval approaches to multi-hop retrieval treat the problem as iterative document retrieval. In particular, at each hop, they reformulate the query representation to account for previous retrieval results so that it can retrieve different documents, usually by concatenating the query and the top-k retrieved documents. However, such an approach is limited since as the number of hops increases, the reformulated query gets longer, and it gradually becomes incapable of containing all the information in a query vector. Overcoming this limitation is a crucial direction for future work.

9.2.4 Improving domain adaptation in KGQA with neural retrieval

Improving the underlying retriever In Chapter 8, we treated the relation prediction task as a retrieval problem and employed a neural retriever with LSTM-based encoders. In preliminary experiments, we found that replacing the encoders with BERT yielded better results in a pure zero-shot setting but did not outperform LSTM-based when synthetic queries from the unseen domains were available during training. A possible future direction can be exploring the reasons for witnessing such a result more in-depth and trying to apply state-of-the-art dense retrievers or cross-encoders to the relation prediction task.

Improving query generation In Chapter 8, we showed that the effectiveness of our proposed KGSQA can be restricted by the quality of the synthetically generated queries. With the recent advancements in LLMs, it would be interesting to explore to what extent LLMs could be used to create queries for the new domains.

Bibliography

[1] J. Allan, W. B. Croft, A. Moffat, and M. Sanderson. Frontiers, challenges, and opportunities for information retrieval: Report from SWIRL 2012 the second strategic workshop on information retrieval in lorne. *SIGIR Forum*, 46(1):2–32, 2012. (Cited on pages 7 and 87.)

[2] A. Asai, K. Hashimoto, H. Hajishirzi, R. Socher, and C. Xiong. Learning to retrieve reasoning paths over wikipedia graph for question answering. In *8th International Conference on Learning Representations, ICLR*, 2020. (Cited on pages 88 and 89.)

[3] A. Atreya and C. Elkan. Latent semantic indexing (lsi) fails for TREC collections. *ACM SIGKDD Explorations Newsletter*, 12(2):5–10, 2011. (Cited on page 2.)

[4] L. Azzopardi, M. Girolami, and C. Van Rijsbergen. Topic based language models for ad hoc information retrieval. In *2004 IEEE International Joint Conference on Neural Networks (IEEE Cat. No. 04CH37541)*, volume 4, pages 3281–3286. IEEE, 2004. (Cited on page 2.)

[5] A. Baevski, Y. Zhou, A. Mohamed, and M. Auli. wav2vec 2.0: A framework for self-supervised learning of speech representations. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020*, 2020. (Cited on pages 64 and 76.)

[6] E. Bassani. ranx: A blazing-fast python library for ranking evaluation and comparison. In *European Conference on Information Retrieval*, pages 259–264. Springer, 2022. (Cited on page 29.)

[7] E. Bastianelli, A. Vanzo, P. Swietojanski, and V. Rieser. Slurp: A spoken language understanding resource package. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 7252–7262, 2020. (Cited on page 61.)

[8] S. Bhargav, G. Sidiropoulos, and E. Kanoulas. 'It's on the tip of my tongue': A new dataset for known-item retrieval. In *WSDM '22: The Fifteenth ACM International Conference on Web Search and Data Mining*. ACM, 2022. (Cited on page 12.)

[9] J. Bitton and Z. Papakipos. Augly: A data augmentations library for audio, image, text, and video. `https://github.com/facebookresearch/AugLy`, 2021. (Cited on page 19.)

[10] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: A collaboratively created graph database for structuring human knowledge. In *Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, 2008. (Cited on pages 105 and 112.)

[11] A. Bordes, N. Usunier, S. Chopra, and J. Weston. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075*, 2015. (Cited on pages 105, 112, and 119.)

[12] H. Bota, K. Zhou, and J. M. Jose. Playing your cards right: The effect of entity cards on search behaviour and workload. In *Proceedings of the 2016 ACM on Conference on Human Information Interaction and Retrieval*, pages 131–140. ACM, 2016. (Cited on page 105.)

[13] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Computer networks and ISDN systems*, 30(1-7):107–117, 1998. (Cited on page 1.)

[14] C. J. Burges. From RankNET to LambdaRank to LambdaMART: An overview. *Learning*, 11(23-581): 81, 2010. (Cited on page 2.)

[15] D. Carmel and E. Yom-Tov. *Estimating the Query Difficulty for Information Retrieval*. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool Publishers, 2010. (Cited on page 41.)

[16] O. Chapelle, Y. Chang, and T.-Y. Liu. Future directions in learning to rank. In *Proceedings of the Learning to Rank Challenge*, pages 91–100. PMLR, 2011. (Cited on page 2.)

[17] C. Chen, P. Zhang, H. Zhang, J. Dai, Y. Yi, H. Zhang, and Y. Zhang. Deep learning on computational-resource-limited platforms: A survey. *Mobile Information Systems*, 2020(1):8454327, 2020. (Cited on page 3.)

[18] C. Chen, J. Zhang, Y. Xu, L. Chen, J. Duan, Y. Chen, S. Tran, B. Zeng, and T. Chilimbi. Why do we need large batchsizes in contrastive learning? a gradient-bias perspective. *Advances in Neural Information Processing Systems*, 35:33860–33875, 2022. (Cited on page 101.)

[19] D. Chen, A. Fisch, J. Weston, and A. Bordes. Reading Wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879, 2017. (Cited on pages 88 and 89.)

[20] C. Chu and R. Wang. A survey of domain adaptation for neural machine translation. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 1304–1319, 2018. (Cited on pages 106 and 110.)

[21] S. Chun, S. J. Oh, R. S. de Rezende, Y. Kalantidis, and D. Larlus. Probabilistic embeddings for cross-modal retrieval. In *IEEE Conference on Computer Vision and Pattern Recognition, (CVPR)*,

2021. (Cited on page 36.)

[22] Y. Chung, C. Zhu, and M. Zeng. SPLAT: speech-language joint pre-training for spoken language understanding. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 1897–1907. Association for Computational Linguistics, 2021. (Cited on page 78.)

[23] J. H. Clark, E. Choi, M. Collins, D. Garrette, T. Kwiatkowski, V. Nikolaev, and J. Palomaki. TyDi QA: A benchmark for information-seeking question answering in typologically diverse languages. *Transactions of the Association for Computational Linguistics*, 8:454–470, 2020. (Cited on page 97.)

[24] D. Cohen, B. Mitra, O. Lesota, N. Rekabsaz, and C. Eickhoff. Not all relevance scores are equal: Efficient uncertainty and calibration modeling for deep retrieval models. In *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 654–664, 2021. (Cited on page 6.)

[25] D. Cohen, B. Mitra, O. Lesota, N. Rekabsaz, and C. Eickhoff. Not all relevance scores are equal: Efficient uncertainty and calibration modeling for deep retrieval models. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021. (Cited on pages 34 and 36.)

[26] W. W. Cohen, H. Sun, R. A. Hofer, and M. Siegler. Scalable neural methods for reasoning with a symbolic knowledge base. In *International Conference on Learning Representations*, 2020. (Cited on page 120.)

[27] N. Craswell, B. Mitra, E. Yilmaz, D. Campos, and E. M. Voorhees. Overview of the TREC 2019 deep learning track. *arXiv preprint arXiv:2003.07820*, 2020. (Cited on page 40.)

[28] N. Craswell, B. Mitra, E. Yilmaz, and D. Campos. Overview of the TREC 2020 deep learning track. *arXiv preprint arXiv:2102.07662*, 2021. (Cited on page 40.)

[29] Z. Dai and J. Callan. Deeper text understanding for ir with contextual neural language modeling. In *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*, pages 985–988, 2019. (Cited on page 16.)

[30] J. Dalton, C. Xiong, and J. Callan. TREC CAsT 2019: The conversational assistance track overview. *Proceedings of the Twenty-Eighth Text REtrieval Conference, TREC*, 2019. (Cited on page 87.)

[31] V. Dang, M. Bendersky, and W. B. Croft. Two-stage learning to rank for information retrieval. In *Advances in Information Retrieval: 35th European Conference on IR Research, ECIR*, pages 423–434. Springer, 2013. (Cited on page 2.)

[32] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. 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)*, pages 4171–4186, 2019. (Cited on pages 3, 15, 29, 78, and 120.)

[33] L. Dietz, B. Gamari, J. Dalton, and N. Craswell. TREC complex answer retrieval overview. In *Proceedings of the Twenty-Seventh Text REtrieval Conference, TREC*, 2018. (Cited on page 87.)

[34] M. Ding, C. Zhou, Q. Chen, H. Yang, and J. Tang. Cognitive graph for multi-hop reading comprehension at scale. *ACL*, 2019. (Cited on page 89.)

[35] P. K. Donepudi. Voice search technology: an overview. *Engineering International*, 2(2):91–102, 2014. (Cited on page 6.)

[36] L. Dong, J. Mallinson, S. Reddy, and M. Lapata. Learning to paraphrase for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 875–886, 2017. (Cited on page 111.)

[37] H. Elsahar, C. Gravier, and F. Laforest. Zero-shot question generation from knowledge graphs for unseen predicates and entity types. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 218–228, 2018. (Cited on pages 106, 110, 111, 113, 114, 115, 116, 118, and 120.)

[38] F. Faisal, S. Keshava, M. M. I. Alam, and A. Anastasopoulos. SD-QA: spoken dialectal question answering for the real world. In *Findings of the Association for Computational Linguistics: EMNLP 2021*, pages 3296–3315. Association for Computational Linguistics, 2021. (Cited on pages 61, 62, 64, 71, and 74.)

[39] Y. Feng, S. Mehri, M. Eskénazi, and T. Zhao. "none of the above": Measure uncertainty in dialog response retrieval. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL*, 2020. (Cited on page 34.)

[40] J. FitzGerald, C. Hench, C. Peris, S. Mackie, K. Rottmann, A. Sanchez, A. Nash, L. Urbach, V. Kakarala, R. Singh, et al. Massive: A 1m-example multilingual natural language understanding dataset with

51 typologically-diverse languages. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 4277–4302, 2023. (Cited on pages 61 and 64.)

[41] G. W. Furnas, T. K. Landauer, L. M. Gomez, and S. T. Dumais. The vocabulary problem in human-system communication. *Communications of the ACM*, 30(11):964–971, 1987. (Cited on page 1.)

[42] Y. Gal and Z. Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the 33nd International Conference on Machine Learning, (ICML)*, 2016. (Cited on page 36.)

[43] P. Gambhir, A. Dev, and S. Agrawal. A contrastive view of vowel phoneme assessment of hindi, indian english and american english speakers. In *International Conference on Artificial Intelligence and Speech Technology*, pages 182–194. Springer, 2021. (Cited on page 67.)

[44] L. Gao, Z. Dai, and J. Callan. Rethink training of BERT rerankers in multi-stage retrieval pipeline. In *Advances in Information Retrieval: 43rd European Conference on IR Research, ECIR*, pages 280–286. Springer, 2021. (Cited on pages 15 and 16.)

[45] L. Gao, Y. Zhang, J. Han, and J. Callan. Scaling deep contrastive learning batch size under memory limited setup. In *Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021)*, pages 316–321, 2021. (Cited on pages 3, 52, 54, and 101.)

[46] L. Gao, X. Ma, J. Lin, and J. Callan. Tevatron: An efficient and flexible toolkit for neural retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval, (SIGIR)*, 2023. (Cited on page 42.)

[47] L. Gao, X. Ma, J. Lin, and J. Callan. Tevatron: An efficient and flexible toolkit for neural retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 3120–3124, 2023. (Cited on page 29.)

[48] J. Guo, Y. Fan, Q. Ai, and W. B. Croft. A deep relevance matching model for ad-hoc retrieval. In *Proceedings of the 25th ACM international on conference on information and knowledge management*, pages 55–64, 2016. (Cited on page 2.)

[49] S. Guo, J. Zhang, Y. Wang, Q. Zhang, C. Li, and H. Chen. Dsm: Question generation over knowledge base via modeling diverse subgraphs with meta-learner. In *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*, pages 4194–4207, 2022. (Cited on page 120.)

[50] S. Guo, J. Zhang, X. Ke, C. Li, and H. Chen. Diversifying question generation over knowledge base via external natural questions. In *Proceedings of the 2024 Joint International Conference on Computational Linguistics, Language Resources and Evaluation (LREC-COLING 2024)*, pages 5096–5108, 2024. (Cited on page 120.)

[51] V. Gupta, M. Chinnakotla, and M. Shrivastava. Retrieve and re-rank: A simple and effective ir approach to simple question answering over knowledge graphs. In *Proceedings of the First Workshop on Fact Extraction and VERification (FEVER)*, pages 22–27, 2018. (Cited on pages 118 and 119.)

[52] M. Hagen, M. Potthast, M. Gohsen, A. Rathgeber, and B. Stein. A large-scale query spelling correction corpus. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1261–1264, 2017. (Cited on pages 19 and 29.)

[53] K. Han, T. C. Ferreira, and C. Gardent. Generating questions from wikidata triples. In *13th Edition of its Language Resources and Evaluation Conference*, 2022. (Cited on page 120.)

[54] X. Han, Z. Liu, and M. Sun. Neural knowledge acquisition via mutual attention between knowledge graph and text. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 32, 2018. (Cited on page 114.)

[55] C. Hauff. Predicting the effectiveness of queries and retrieval systems. *SIGIR Forum*, 44, 2010. (Cited on page 41.)

[56] B. He and I. Ounis. Inferring query performance using pre-retrieval predictors. In *String Processing and Information Retrieval, SPIRE*, 2004. (Cited on page 41.)

[57] B. He and I. Ounis. Query performance prediction. *Information Systems*, 31(7):585–594, 2006. (Cited on page 41.)

[58] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. (Cited on page 108.)

[59] X. He and D. Golub. Character-level question answering with attention. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 1598–1607, 2016. (Cited on page 119.)

[60] J. Herzig and J. Berant. Decoupling structure and lexicon for zero-shot semantic parsing. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 1619–1629,

2018. (Cited on page 119.)

[61] M. Heuss, D. Cohen, M. Mansoury, M. de Rijke, and C. Eickhoff. Predictive uncertainty-based bias mitigation in ranking. In *Proceedings of the 32nd ACM International Conference on Information and Knowledge Management, (CIKM)*, 2023. (Cited on pages 34 and 36.)

[62] T. Hofmann. Unsupervised learning by probabilistic latent semantic analysis. *Machine learning*, 42: 177–196, 2001. (Cited on page 2.)

[63] S. Hofstätter, S. Althammer, M. Schröder, M. Sertkan, and A. Hanbury. Improving efficient neural ranking models with cross-architecture knowledge distillation. *arXiv preprint arXiv:2010.02666*, 2020. (Cited on pages 39 and 100.)

[64] S. Hofstätter, S. Lin, J. Yang, J. Lin, and A. Hanbury. Efficiently teaching an effective dense retriever with balanced topic aware sampling. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2021. (Cited on pages 37, 41, 49, and 120.)

[65] D. Hoogeveen, K. M. Verspoor, and T. Baldwin. CQADupStack: A benchmark data set for community question-answering research. In *Proceedings of the 20th Australasian Document Computing Symposium (ADCS)*, ADCS '15, 2015. (Cited on page 40.)

[66] W. Hsu, B. Bolte, Y. H. Tsai, K. Lakhotia, R. Salakhutdinov, and A. Mohamed. HuBERT: Self-supervised speech representation learning by masked prediction of hidden units. *IEEE ACM Trans. Audio Speech Lang. Process.*, 29:3451–3460, 2021. (Cited on pages 75 and 78.)

[67] Z. Huang, W. Xu, and K. Yu. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*, 2015. (Cited on page 107.)

[68] D. Hwang, A. Misra, Z. Huo, N. Siddhartha, S. Garg, D. Qiu, K. C. Sim, T. Strohman, F. Beaufays, and Y. He. Large-scale ASR domain adaptation using self- and semi-supervised learning. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2022*, pages 6627–6631. IEEE, 2022. (Cited on page 72.)

[69] T. Joachims. Optimizing search engines using clickthrough data. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 133–142, 2002. (Cited on page 1.)

[70] J. Johnson, M. Douze, and H. Jégou. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, 7(3):535–547, 2019. (Cited on pages 2, 3, 16, 34, 39, 65, and 76.)

[71] P. Jonk, V. de Vries, R. Wever, G. Sidiropoulos, and E. Kanoulas. Natural language processing of aviation occurrence reports for safety management. In *Proceedings of the 32nd European Safety and Reliability Conference*. RPS, 2022. (Cited on page 12.)

[72] E. Kanoulas, E. Yilmaz, R. Mehrotra, B. Carterette, N. Craswell, and P. Bailey. TREC 2017 tasks track overview. In *Proceedings of The Twenty-Sixth Text REtrieval Conference, TREC*, 2017. (Cited on page 87.)

[73] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W.-t. Yih. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, 2020. (Cited on pages 1, 3, 15, 17, 19, 25, 27, 33, 37, 40, 49, 62, 65, 66, 72, 74, 75, 76, 77, 78, 82, 83, 89, 92, 93, 101, and 120.)

[74] O. Khattab and M. Zaharia. Colbert: Efficient and effective passage search via contextualized late interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR conference on research and development in Information Retrieval*, pages 39–48, 2020. (Cited on pages 15, 17, 23, and 120.)

[75] P. Khosla, P. Teterwak, C. Wang, A. Sarna, Y. Tian, P. Isola, A. Maschinot, C. Liu, and D. Krishnan. Supervised contrastive learning. *Advances in neural information processing systems*, 33:18661–18673, 2020. (Cited on pages 26, 27, and 101.)

[76] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014. (Cited on page 114.)

[77] J. Kiseleva, K. Williams, J. Jiang, A. Hassan Awadallah, A. C. Crook, I. Zitouni, and T. Anastasakos. Understanding user satisfaction with intelligent assistants. In *Proceedings of the 2016 ACM on conference on human information interaction and retrieval*, pages 121–130, 2016. (Cited on page 8.)

[78] T. Kwiatkowski, J. Palomaki, O. Redfield, M. Collins, A. Parikh, C. Alberti, D. Epstein, I. Polosukhin, J. Devlin, K. Lee, et al. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 2019. (Cited on pages 3, 17, 63, and 77.)

[79] G. Lample, M. Ballesteros, S. Subramanian, K. Kawakami, and C. Dyer. Neural architectures for named entity recognition. In *Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 260–270, 2016. (Cited on page 107.)

[80] H. Lee, S. Yang, H. Oh, and M. Seo. Generative multi-hop retrieval. In *Proceedings of the 2022*

Conference on Empirical Methods in Natural Language Processing, pages 1417–1436, 2022. (Cited on pages 7 and 101.)

[81] J. Lee, S. Yun, H. Kim, M. Ko, and J. Kang. Ranking paragraphs for improving answer recall in open-domain question answering. In *EMNLP*, 2018. (Cited on page 89.)

[82] K. Lee, M.-W. Chang, and K. Toutanova. Latent retrieval for weakly supervised open domain question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6086–6096, 2019. (Cited on pages 1, 2, and 88.)

[83] N. Levine, H. Roitman, and D. Cohen. An extended relevance model for session search. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 865–868, 2017. (Cited on page 87.)

[84] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7871–7880, 2020. (Cited on page 120.)

[85] C. Li, S. Wu, C. Liu, and H. Lee. Spoken squad: A study of mitigating the impact of speech recognition errors on listening comprehension. In *Interspeech 2018, 19th Annual Conference of the International Speech Communication Association*, pages 3459–3463. ISCA, 2018. (Cited on pages 61, 64, and 74.)

[86] G. Li, J. Wang, Y. Zheng, and M. J. Franklin. Crowdsourced data management: A survey. *IEEE Transactions on Knowledge and Data Engineering*, 28(9):2296–2319, 2016. (Cited on page 3.)

[87] H. Li, J. Xu, et al. Semantic matching in search. *Foundations and Trends® in Information Retrieval*, 7 (5):343–469, 2014. (Cited on page 1.)

[88] H. Li, K. Ota, M. Dong, A. V. Vasilakos, and K. Nagano. Multimedia processing pricing strategy in gpu-accelerated cloud computing. *IEEE Transactions on Cloud Computing*, 8(4):1264–1273, 2017. (Cited on page 4.)

[89] X. Li, Z. Zhou, S. Dalmia, A. W. Black, and F. Metze. SANTLR: speech annotation toolkit for low resource languages. In *Interspeech 2019, 20th Annual Conference of the International Speech Communication Association*, pages 3681–3682. ISCA, 2019. (Cited on page 64.)

[90] Y. Li, Z. Liu, C. Xiong, and Z. Liu. More robust dense retrieval with contrastive dual learning. In *Proceedings of the 2021 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 287–296, 2021. (Cited on page 27.)

[91] S. Lin, J. Yang, and J. Lin. In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In *Proceedings of the 6th Workshop on Representation Learning for NLP, RepL4NLP@ACL-IJCNLP*, 2021. (Cited on pages 49 and 57.)

[92] S. Lindemann. Who speaks "broken english"? us undergraduates' perceptions of non-native english 1. *International Journal of Applied Linguistics*, 15(2):187–212, 2005. (Cited on page 67.)

[93] T.-Y. Liu et al. Learning to rank for information retrieval. *Foundations and Trends® in Information Retrieval*, 3(3):225–331, 2009. (Cited on page 2.)

[94] Y. Luan, J. Eisenstein, K. Toutanova, and M. Collins. Sparse, dense, and attentional representations for text retrieval. *Transactions of the Association for Computational Linguistics*, 9:329–345, 2021. (Cited on pages 3 and 15.)

[95] D. Lukovnikov, A. Fischer, J. Lehmann, and S. Auer. Neural network-based question answering over knowledge graphs on word and character level. In *Proceedings of the 26th international conference on World Wide Web*, pages 1211–1220, 2017. (Cited on page 119.)

[96] X. Ma, C. dos Santos, and A. O. Arnold. Contrastive fine-tuning improves robustness for neural rankers. In *Findings of the Association for Computational Linguistics: ACL-IJCNLP 2021*, pages 570–582, 2021. (Cited on page 16.)

[97] M. Maia, S. Handschuh, A. Freitas, B. Davis, R. McDermott, M. Zarrouk, and A. Balahur. Www'18 open challenge: Financial opinion mining and question answering. In *Companion Proceedings of the The Web Conference 2018*, 2018. (Cited on page 40.)

[98] Y. Mansour, M. Mohri, and A. Rostamizadeh. Domain adaptation: Learning bounds and algorithms. In *22nd Conference on Learning Theory, COLT 2009*, 2009. (Cited on page 106.)

[99] T. Mao, Y. Khassanov, V. T. Pham, H. Xu, H. Huang, and E. S. Chng. Approaches to improving recognition of underrepresented named entities in hybrid ASR systems. In *12th International Symposium on Chinese Spoken Language Processing, ISCSLP 2021*, pages 1–5. IEEE, 2021. (Cited on page 72.)

[100] E. P. Markatos. On caching search engine query results. *Computer Communications*, 24(2):137–143, 2001. (Cited on page 3.)

[101] M. Małkiński and J. Mańdziuk. Multi-label contrastive learning for abstract visual reasoning. *IEEE Transactions on Neural Networks and Learning Systems*, pages 1–13, 2022. doi: 10.1109/TNNLS.

2022.3185949. (Cited on page 26.)
[102] N. Mckenna and P. Sen. Kgqa without retraining. In *Proceedings of The Fourth Workshop on Simple and Efficient Natural Language Processing (SustaiNLP)*, pages 212–218, 2023. (Cited on page 120.)
[103] C. Meng, N. Arabzadeh, M. Aliannejadi, and M. de Rijke. Query performance prediction: From ad-hoc to conversational search. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 2583–2593, 2023. (Cited on page 42.)
[104] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 26, 2013. (Cited on pages 109 and 114.)
[105] S. Min, V. Zhong, L. Zettlemoyer, and H. Hajishirzi. Multi-hop reading comprehension through question decomposition and rescoring. *ACL*, 2019. (Cited on page 89.)
[106] M. Mintz, S. Bills, R. Snow, and D. Jurafsky. Distant supervision for relation extraction without labeled data. 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*, pages 1003–1011, 2009. (Cited on page 112.)
[107] B. Mitra, E. Nalisnick, N. Craswell, and R. Caruana. A dual embedding space model for document ranking. *arXiv preprint arXiv:1602.01137*, 2016. (Cited on page 2.)
[108] B. Mitra, N. Craswell, et al. An introduction to neural information retrieval. *Foundations and Trends® in Information Retrieval*, 13(1):1–126, 2018. (Cited on page 1.)
[109] V. Mnih, A. P. Badia, M. Mirza, A. Graves, T. Lillicrap, T. Harley, D. Silver, and K. Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937. PMLR, 2016. (Cited on page 94.)
[110] S. Mohammed, P. Shi, and J. Lin. Strong baselines for simple question answering over knowledge graphs with and without neural networks. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 291–296, 2018. (Cited on pages 105, 110, 113, and 119.)
[111] N. S. Moosavi, I. Gurevych, A. Fan, T. Wolf, Y. Hou, A. Marasović, and S. Ravi, editors. *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing*, 2021. Association for Computational Linguistics. (Cited on page 4.)
[112] J. Morris, E. Lifland, J. Y. Yoo, J. Grigsby, D. Jin, and Y. Qi. Textattack: A framework for adversarial attacks, data augmentation, and adversarial training in nlp. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 119–126, 2020. (Cited on pages 28 and 29.)
[113] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng. MS MARCO: A human generated machine reading comprehension dataset. In *CoCo@ NIPS*, 2016. (Cited on pages 3, 17, 29, 40, 63, and 77.)
[114] Y. Nie, S. Wang, and M. Bansal. Revealing the importance of semantic retrieval for machine reading at scale. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2553–2566, 2019. (Cited on pages 88, 89, and 97.)
[115] R. Nogueira and K. Cho. Passage re-ranking with BERT. *arXiv preprint arXiv:1901.04085*, 2019. (Cited on pages 3, 16, and 91.)
[116] A. Oord, Y. Li, and O. Vinyals. Representation learning with contrastive predictive coding. *arXiv preprint arXiv:1807.03748*, 2018. (Cited on page 3.)
[117] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009. (Cited on page 106.)
[118] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur. Librispeech: An ASR corpus based on public domain audio books. In *2015 IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2015*, pages 5206–5210. IEEE, 2015. (Cited on page 76.)
[119] T. Pellissier Tanon, D. Vrandečić, S. Schaffert, T. Steiner, and L. Pintscher. From Freebase to Wikidata: The great migration. In *Proceedings of the 25th international conference on world wide web*, pages 1419–1428. International World Wide Web Conferences Steering Committee, 2016. (Cited on pages 105 and 106.)
[120] M. Petrochuk and L. Zettlemoyer. Simplequestions nearly solved: A new upperbound and baseline approach. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 554–558, 2018. (Cited on pages 105, 107, 108, 110, 113, 116, 118, and 119.)
[121] A. Pradhan, K. Mehta, and L. Findlater. "Accessibility came by accident": Use of voice-controlled intelligent personal assistants by people with disabilities. In *Proceedings of the 2018 CHI Conference*

on human factors in computing systems, pages 1–13, 2018. (Cited on pages 6 and 61.)

[122] D. Pruthi, B. Dhingra, and Z. C. Lipton. Combating adversarial misspellings with robust word recognition. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 5582–5591, 2019. (Cited on page 16.)

[123] P. Qi, X. Lin, L. Mehr, Z. Wang, and C. D. Manning. Answering complex open-domain questions through iterative query generation. In *2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2019. (Cited on pages 88 and 89.)

[124] Y. Qu, Y. Ding, J. Liu, K. Liu, R. Ren, W. X. Zhao, D. Dong, H. Wu, and H. Wang. RocketQA: An optimized training approach to dense passage retrieval for open-domain question answering. In K. Toutanova, A. Rumshisky, L. Zettlemoyer, D. Hakkani-Tür, I. Beltagy, S. Bethard, R. Cotterell, T. Chakraborty, and Y. Zhou, editors, *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT*, pages 5835–5847. Association for Computational Linguistics, 2021. (Cited on pages 17, 74, and 82.)

[125] A. Radford, J. W. Kim, T. Xu, G. Brockman, C. McLeavey, and I. Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518. PMLR, 2023. (Cited on pages 83 and 127.)

[126] A. Ravichander, S. Dalmia, M. Ryskina, F. Metze, E. Hovy, and A. W. Black. NoiseQA: Challenge set evaluation for user-centric question answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 2976–2992, 2021. (Cited on pages 61, 62, 64, 71, and 74.)

[127] 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. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 2825–2835, 2021. (Cited on pages 1, 3, and 62.)

[128] S. Robertson, H. Zaragoza, et al. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends® in Information Retrieval*, 3(4):333–389, 2009. (Cited on pages 1 and 2.)

[129] S. E. Robertson. The probability ranking principle in IR. *Journal of documentation*, 33(4):294–304, 1977. (Cited on page 1.)

[130] D. E. Rose and D. Levinson. Understanding user goals in web search. In *Proceedings of the 13th international conference on World Wide Web*, pages 13–19, 2004. (Cited on page 1.)

[131] N. Sadat Moosavi, I. Gurevych, Y. Hou, G. Kim, Y. J. Kim, T. Schuster, and A. Agrawal, editors. *Proceedings of The Fourth Workshop on Simple and Efficient Natural Language Processing (SustaiNLP)*, Toronto, Canada (Hybrid), July 2023. Association for Computational Linguistics. (Cited on page 4.)

[132] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval. *Information processing & management*, 24(5):513–523, 1988. (Cited on page 1.)

[133] V. Sanh, L. Debut, J. Chaumond, and T. Wolf. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. (Cited on page 37.)

[134] H. Scells, S. Zhuang, and G. Zuccon. Reduce, reuse, recycle: Green information retrieval research. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 2825–2837, 2022. (Cited on page 4.)

[135] J. Schalkwyk, D. Beeferman, F. Beaufays, B. Byrne, C. Chelba, M. Cohen, M. Kamvar, and B. Strope. "your word is my command": Google search by voice: A case study. *Advances in Speech Recognition: Mobile Environments, Call Centers and Clinics*, pages 61–90, 2010. (Cited on page 6.)

[136] Y. Shen, X. He, J. Gao, L. Deng, and G. Mesnil. Learning semantic representations using convolutional neural networks for web search. In *Proceedings of the 23rd international conference on world wide web*, pages 373–374, 2014. (Cited on page 2.)

[137] F. H. Shezan, H. Hu, J. Wang, G. Wang, and Y. Tian. Read between the lines: An empirical measurement of sensitive applications of voice personal assistant systems. In *Proceedings of the Web Conference 2020*, pages 1006–1017, 2020. (Cited on page 6.)

[138] G. Sidiropoulos and E. Kanoulas. Multimodal dense passage retrieval for open-domain spoken question answering. *Under submission*. (Cited on page 11.)

[139] G. Sidiropoulos and E. Kanoulas. Analysing the robustness of dual encoders for dense retrieval against misspellings. In *SIGIR '22: The 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*. ACM, 2022. (Cited on pages 10, 25, 27, 42, 62, 65, 74, 76, 77, and 80.)

[140] G. Sidiropoulos and E. Kanoulas. Improving the robustness of dense retrievers against typos via

multi-positive contrastive learning. In *Advances in Information Retrieval - 46th European Conference on Information Retrieval, ECIR, Proceedings, Part III*. Springer, 2024. (Cited on pages 10 and 42.)

[141] G. Sidiropoulos, S. Bhargav, P. Eustratiadis, and E. Kanoulas. Multivariate dense retrieval: A reproducibility study under a memory-limited setup. *Under submission*. (Cited on page 11.)

[142] G. Sidiropoulos, N. Voskarides, and E. Kanoulas. Knowledge graph simple question answering for unseen domains. In *Conference on Automated Knowledge Base Construction, AKBC*, 2020. (Cited on page 12.)

[143] G. Sidiropoulos, N. Voskarides, S. Vakulenko, and E. Kanoulas. Combining lexical and dense retrieval for computationally efficient multi-hop question answering. In *Proceedings of the Second Workshop on Simple and Efficient Natural Language Processing, SustaiNLP@EMNLP 2021*. Association for Computational Linguistics, 2021. (Cited on pages 11 and 15.)

[144] G. Sidiropoulos, S. Vakulenko, and E. Kanoulas. On the impact of speech recognition errors in passage retrieval for spoken question answering. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*. ACM, 2022. (Cited on pages 11, 25, 72, 73, 74, 76, and 77.)

[145] R. Snow, B. O'connor, D. Jurafsky, and A. Y. Ng. Cheap and fast–but is it good? evaluating non-expert annotations for natural language tasks. In *Proceedings of the 2008 conference on empirical methods in natural language processing*, pages 254–263, 2008. (Cited on page 3.)

[146] A. Sorokin and D. Forsyth. Utility data annotation with Amazon Mechanical Turk. In *2008 IEEE computer society conference on computer vision and pattern recognition workshops*, pages 1–8. IEEE, 2008. (Cited on page 3.)

[147] N. Spasojevic, P. Bhargava, and G. Hu. Dawt: Densely annotated wikipedia texts across multiple languages. In *Proceedings of the 26th International Conference on World Wide Web Companion*, pages 1655–1662, 2017. (Cited on page 113.)

[148] E. Strubell, A. Ganesh, and A. McCallum. Energy and policy considerations for modern deep learning research. In *Proceedings of the AAAI conference on artificial intelligence*, volume 34, pages 13693–13696, 2020. (Cited on page 4.)

[149] Y. Su and X. Yan. Cross-domain semantic parsing via paraphrasing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1235–1246, 2017. (Cited on page 119.)

[150] L. Sun, K. Hashimoto, W. Yin, A. Asai, J. Li, P. Yu, and C. Xiong. Adv-bert: Bert is not robust on misspellings! generating nature adversarial samples on bert. *arXiv preprint arXiv:2003.04985*, 2020. (Cited on page 16.)

[151] A. Talmor and J. Berant. The web as a knowledge-base for answering complex questions. In *NAACL-HLT*, 2018. (Cited on page 90.)

[152] C. C. Tan, B. Sheng, H. Wang, and Q. Li. Microsearch: When search engines meet small devices. In *Pervasive Computing: 6th International Conference*, pages 93–110. Springer, 2008. (Cited on page 3.)

[153] P. Tasawong, W. Ponwitayarat, P. Limkonchotiwat, C. Udomcharoenchaikit, E. Chuangsuwanich, and S. Nutanong. Typo-robust representation learning for dense retrieval. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 1106–1115, 2023. (Cited on pages 25, 26, 28, and 29.)

[154] N. Thakur, N. Reimers, A. Rücklé, A. Srivastava, and I. Gurevych. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*, 2021. (Cited on pages 3 and 62.)

[155] M. Trabelsi, Z. Chen, B. D. Davison, and J. Heflin. Neural ranking models for document retrieval. *Information Retrieval Journal*, 24(6):400–444, 2021. (Cited on page 2.)

[156] K. Tsukada. An acoustic comparison between american english and australian english vowels. In *INTERSPEECH*, pages 2257–2260, 2002. (Cited on page 67.)

[157] F. Türe and O. Jojic. No need to pay attention: Simple recurrent neural networks work! In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2866–2872, 2017. (Cited on pages 108, 118, and 119.)

[158] S. Vakulenko, S. Longpre, Z. Tu, and R. Anantha. Question rewriting for conversational question answering. In *Proceedings of the 14th ACM International Conference on Web Search and Data Mining*, pages 355–363, 2021. (Cited on page 87.)

[159] C. Van Gysel and M. de Rijke. Pytrec_eval: An extremely fast python interface to trec_eval. In *SIGIR*. ACM, 2018. (Cited on page 42.)

[160] L. Vilnis and A. McCallum. Word representations via gaussian embedding. In *3rd International Conference on Learning Representations (ICLR '15)*, 2015. (Cited on page 34.)

[161] E. Voorhees, T. Alam, S. Bedrick, D. Demner-Fushman, W. R. Hersh, K. Lo, K. Roberts, I. Soboroff, and L. L. Wang. TREC-COVID: constructing a pandemic information retrieval test collection. In *ACM SIGIR Forum*, volume 54, 2021. (Cited on page 40.)

[162] N. Voskarides, E. Meij, M. Tsagkias, M. de Rijke, and W. Weerkamp. Learning to explain entity relationships in knowledge graphs. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 564–574. Association for Computational Linguistics, 2015. (Cited on page 112.)

[163] N. Voskarides, D. Li, P. Ren, E. Kanoulas, and M. de Rijke. Query resolution for conversational search with limited supervision. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 921–930, 2020. (Cited on page 87.)

[164] D. Wadden, S. Lin, K. Lo, L. L. Wang, M. van Zuylen, A. Cohan, and H. Hajishirzi. Fact or fiction: Verifying scientific claims. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 2020. (Cited on page 40.)

[165] H. Wang, S. Dong, Y. Liu, J. Logan, A. K. Agrawal, and Y. Liu. ASR error correction with augmented transformer for entity retrieval. In H. Meng, B. Xu, and T. F. Zheng, editors, *Interspeech 2020, 21st Annual Conference of the International Speech Communication Association*, pages 1550–1554. ISCA, 2020. (Cited on page 72.)

[166] J. Wang and J. Zhu. Portfolio theory of information retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009. (Cited on page 34.)

[167] F. Warburg, M. Jørgensen, J. Civera, and S. Hauberg. Bayesian triplet loss: Uncertainty quantification in image retrieval. In *IEEE/CVF International Conference on Computer Vision, (ICCV)*, 2021. (Cited on pages 36 and 47.)

[168] F. Warburg, M. Miani, S. Brack, and S. Hauberg. Bayesian metric learning for uncertainty quantification in image retrieval. In A. Oh, T. Naumann, A. Globerson, K. Saenko, M. Hardt, and S. Levine, editors, *Advances in Neural Information Processing Systems 36: Annual Conference on Neural Information Processing Systems 2023, NeurIPS 2023*, 2023. (Cited on page 6.)

[169] F. Warburg, M. Miani, S. Brack, and S. Hauberg. Bayesian metric learning for uncertainty quantification in image retrieval. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2023. (Cited on pages 36 and 47.)

[170] X. Wei and W. B. Croft. Lda-based document models for ad-hoc retrieval. In *Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 178–185, 2006. (Cited on page 2.)

[171] P. Wu, S. Huang, R. Weng, Z. Zheng, J. Zhang, X. Yan, and J. Chen. Learning representation mapping for relation detection in knowledge base question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6130–6139, 2019. (Cited on pages 107, 114, 118, and 119.)

[172] Z. Wu, Y. Xiong, S. X. Yu, and D. Lin. Unsupervised feature learning via non-parametric instance discrimination. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 3733–3742, 2018. (Cited on page 92.)

[173] G. Xiong, J. Bao, W. Zhao, Y. Wu, and X. He. Autoqgs: Auto-prompt for low-resource knowledge-based question generation from sparql. In *Proceedings of the 31st ACM International Conference on Information & Knowledge Management*, pages 2250–2259, 2022. (Cited on page 120.)

[174] L. Xiong, C. Xiong, Y. Li, K. Tang, J. Liu, P. N. Bennett, J. Ahmed, and A. Overwijk. Approximate nearest neighbor negative contrastive learning for dense text retrieval. In *9th International Conference on Learning Representations, ICLR 2021*. OpenReview.net, 2021. (Cited on page 83.)

[175] W. Xiong, X. Li, S. Iyer, J. Du, P. Lewis, W. Y. Wang, Y. Mehdad, S. Yih, S. Riedel, D. Kiela, et al. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*, 2020. (Cited on pages 7, 88, 89, 91, 92, 93, 94, 97, and 101.)

[176] H. Yang, D. Guan, and S. Zhang. The query change model: Modeling session search as a markov decision process. *ACM Transactions on Information Systems (TOIS)*, 33(4):1–33, 2015. (Cited on page 87.)

[177] P. Yang, H. Fang, and J. Lin. Anserini: Enabling the use of lucene for information retrieval research. In *Proceedings of the 40th international ACM SIGIR conference on research and development in information retrieval*, pages 1253–1256, 2017. (Cited on pages 65 and 91.)

[178] W. Yang, Y. Xie, A. Lin, X. Li, L. Tan, K. Xiong, M. Li, and J. Lin. End-to-end open-domain question answering with BERTserini. In W. Ammar, A. Louis, and N. Mostafazadeh, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics*

(Demonstrations), pages 72–77. Association for Computational Linguistics, 2019. (Cited on page 88.)

[179] Z. Yang, P. Qi, S. Zhang, Y. Bengio, W. Cohen, R. Salakhutdinov, and C. D. Manning. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, 2018. (Cited on pages 3, 87, 90, and 91.)

[180] X. Yi and J. Allan. A comparative study of utilizing topic models for information retrieval. In *Advances in Information Retrieval: 31th European Conference on IR Research, ECIR*, pages 29–41. Springer, 2009. (Cited on page 2.)

[181] W.-t. Yih, M.-W. Chang, X. He, and J. Gao. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *ACL-IJCNLP*, pages 1321–1331. ACL, 2015. (Cited on page 105.)

[182] W. Yin, M. Yu, B. Xiang, B. Zhou, and H. Schütze. Simple question answering by attentive convolutional neural network. In N. Calzolari, Y. Matsumoto, and R. Prasad, editors, *COLING 2016, 26th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers*, pages 1746–1756. ACL, 2016. (Cited on page 119.)

[183] C. You, N. Chen, and Y. Zou. Knowledge distillation for improved accuracy in spoken question answering. In *IEEE International Conference on Acoustics, Speech and Signal Processing, ICASSP 2021*, pages 7793–7797. IEEE, 2021. (Cited on page 74.)

[184] M. Yu, W. Yin, K. S. Hasan, C. dos Santos, B. Xiang, and B. Zhou. Improved neural relation detection for knowledge base question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 571–581, 2017. (Cited on pages 107, 108, 113, 114, 116, and 119.)

[185] T. Yu, R. Zhang, K. Yang, M. Yasunaga, D. Wang, Z. Li, J. Ma, I. Li, Q. Yao, S. Roman, et al. Spider: A large-scale human-labeled dataset for complex and cross-domain semantic parsing and text-to-sql task. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 3911–3921, 2018. (Cited on page 119.)

[186] L. A. Zadeh. From search engines to question answering systems—the problems of world knowledge, relevance, deduction and precisiation. In *Capturing intelligence*, volume 1, pages 163–210. Elsevier, 2006. (Cited on page 8.)

[187] H. Zamani and M. Bendersky. Multivariate representation learning for information retrieval. In *Proceedings of the 46th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 163–173, 2023. (Cited on pages 6, 33, 34, 39, 40, 41, 42, 45, 48, 53, 55, 56, 57, 124, 141, and 143.)

[188] H. Zeng, H. Zamani, and V. Vinay. Curriculum learning for dense retrieval distillation. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1979–1983, 2022. (Cited on pages 3, 37, 39, 41, and 57.)

[189] L. Zhang and Y. Rui. Image search—from thousands to billions in 20 years. *ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)*, 9(1s):1–20, 2013. (Cited on page 6.)

[190] R. Zhang, T. Yu, H. Er, S. Shim, E. Xue, X. V. Lin, T. Shi, C. Xiong, R. Socher, and D. Radev. Editing-based sql query generation for cross-domain context-dependent questions. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 5341–5352, 2019. (Cited on page 119.)

[191] W. Zhang, W. Hua, and K. Stratos. Entqa: Entity linking as question answering. In *The Tenth International Conference on Learning Representations, ICLR*, 2022. (Cited on page 126.)

[192] W. Zhao, T. Chung, A. Goyal, and A. Metallinou. Simple question answering with subgraph ranking and joint-scoring. 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)*, pages 324–334, 2019. (Cited on page 118.)

[193] Y. Zhao, F. Scholer, and Y. Tsegay. Effective pre-retrieval query performance prediction using similarity and variability evidence. In *European Conference on IR Research, ECIR*, 2008. (Cited on page 41.)

[194] F. Zhu, W. Lei, C. Wang, J. Zheng, S. Poria, and T. Chua. Retrieving and reading: A comprehensive survey on open-domain question answering. *CoRR*, abs/2101.00774, 2021. (Cited on pages 71 and 74.)

[195] J. Zhu, J. Wang, I. J. Cox, and M. J. Taylor. Risky business: modeling and exploiting uncertainty in information retrieval. In *ACM SIGIR Conference on Research and Development in Information Retrieval*, 2009. (Cited on page 34.)

[196] S. Zhuang and G. Zuccon. Dealing with typos for BERT-based passage retrieval and ranking. In

Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, pages 2836–2842, 2021. (Cited on pages 3, 16, 18, 19, 62, 74, and 76.)

[197] S. Zhuang and G. Zuccon. CharacterBERT and self-teaching for improving the robustness of dense retrievers on queries with typos. In *Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 1444–1454, 2022. (Cited on pages 25, 29, 31, 47, 62, 65, 66, 74, and 77.)

Summary

Over the past years, the emergence of neural retrievers has brought significant advancements in various IR tasks. Search engines have used such models in an attempt to bridge the semantic gap between queries and the corpus. In this thesis, we focus on developing robust and effective neural retrievers to support various functionalities of modern search engines. In detail, to fulfill the user information needs, modern search engines need to: (i) be able to deal with queries regardless of their complexity (i.e., single-hop or multi-hop queries) and frequency (popular, uncommon, or previously unseen), (ii) not be sensitive to errors that might appear on the query (e.g., typos, ASR errors, etc.), (iii) access unstructured and structured knowledge sources, (iv) have QA functionalities to provide direct answers to the given queries, and finally (v) support multiple means via which the users can interact with the system (e.g., keyboard, voice, etc.).

In the first part, we study how to increase the robustness of dense retrievers when the query contains typos for the task of ad-hoc retrieval. While dense retrievers achieve high retrieval performance on typo-free queries, previous studies have shown a significant decrease when dealing with typoed queries. In our first study, in order to increase robustness, we propose an alternative training setup for the dense retriever that combines data augmentation with contrastive learning, aiming to maximize the agreement between the original typo-free query and its typoed variation. Even though our method increases robustness, the contrastive loss–that is employed–assumes a single positive sample and a set of negative ones per anchor, hence not fully utilizing the multiple positives (e.g., multiple synthetic typoed queries). This leads us to our second study, in which we propose to enhance existing typo-robust dense retrievers by incorporating contrastive learning that accounts for multiple positives and negatives. Besides learning better representations for the noisy text, uncertainty estimation can also be used for robustness. The multivariate representation learning framework proposed by Zamani and Bendersky [187] is the first method to work in the direction of modeling uncertainty in dense retrieval. Thus, in our third and last study, we explore whether the multivariate representation learning framework can capture uncertainty in queries containing typos.

In the second part of this thesis, we move to a different research theme and study how to improve the robustness of speech-based search with neural retrieval. Here, the queries are voiced, and the documents are in textual form. Our first study centers on ASR-Retriever pipeline approaches, where the spoken queries are transcribed using an ASR model and then passed through the text retriever. Specifically, we aim to assess the robustness of dense retrievers when dealing with queries that contain transcription errors. We demonstrate that current state-of-the-art dense retrievers are not inherently robust on queries with ASR errors. To address this, we propose training with transcriptions from synthetically generated spoken queries to increase robustness. Two prominent shortcomings of ASR-retriever pipelines are that the ASR propagates its errors to the downstream retriever ASR-Retriever pipelines and that training an effective ASR model requires a large amount of annotated speech. This leads us to our second study, where we explore how to alleviate such limitations of the ASR-retriever pipelines. We propose a multimodal, ASR-free dense retriever that can work directly on the spoken query.

In the third part of this thesis, we focus on complex queries (also known as multi-hop queries) that cannot be resolved in a single document. Such queries require multi-

hop retrieval to gather multiple documents to provide adequate evidence collectively. Specifically, we study the effectiveness of dense retrievers for multi-hop retrieval on limited computing resources. We showcase the performance of a decrease in limited resources and propose a computationally efficient hybrid retriever that combines lexical and dense retrieval.

In this thesis's fourth and last part, we focus on a new research topic: improving domain adaptation in KGQA with neural retrieval. We focus on a scenario where new domains (i.e., subgraphs) covering unseen relations and entities are added to the KG–where there are no available query-answer pairs during training time–and the KGQA system follows a pipeline approach consisting of entity mention detection, entity candidate generation, relation prediction, and answer selection. We focus on relation prediction since we identified it does not scale to new domains and we set to increase its robustness over previously unseen domains. We start by formulating relation prediction as a retrieval problem using the textual label to represent the relation–contrary to most works that treat it as classification–allowing us to represent any relation during inference time. Furthermore, we propose to use a neural retriever in combination with data augmentation to augment the training set with synthetically generated queries for the new domains.

Samenvatting

In de afgelopen jaren heeft de opkomst van *neural retrievers* aanzienlijke vooruitgang gebracht in verschillende IR-taken. Zoekmachines hebben dergelijke modellen gebruikt om de semantische kloof tussen zoekopdrachten en het tekstcorpus te overbruggen. In deze scriptie richten we ons op het ontwikkelen van robuuste en effectieve *neural retrievers* om diverse functionaliteiten van moderne zoekmachines te ondersteunen. Om te voldoen aan de informatiebehoeften van gebruikers moeten moderne zoekmachine: (i) in staat zijn om te gaan met zoekopdrachten ongeacht hun complexiteit (d.w.z. *single-hop* of *multi-hop* zoekopdrachten) en frequentie (populair, ongebruikelijk of nog nooit eerder gezien), (ii) niet gevoelig zijn voor fouten die in de zoekopdracht kunnen voorkomen (bijv. typfouten, fouten op basis van automatische spraakherkenning, enz.), (iii) toegang hebben tot ongestructureerde en gestructureerde kennisbronnen, (iv) vraag- en anntwoordfunctionaliteiten hebben om directe antwoorden op de gegeven zoekopdrachten te bieden, en tenslotte (v) meerdere middelen ondersteunen waarmee gebruikers met het systeem kunnen communiceren (bijv. toetsenbord, spraak, enz.).

In het eerste deel onderzoeken we hoe we de robuustheid van *dense retrievers* kunnen vergroten wanneer de zoekopdracht typefouten bevat voor de taak van een *ad-hoc retrieval*. Hoewel *dense retrievers* hoge prestaties behalen bij typefoutvrije zoekopdrachten, hebben eerdere studies een significante afname aangetoond bij het omgaan met zoekopdrachten met typefouten. In onze eerste studie stellen we een alternatieve trainingsopzet voor de *dense retriever* voor om de robuustheid te vergroten, gegevensvergroting combineert met *contrastive learning* met als doel de overeenstemming tussen de originele typefoutvrije zoekopdracht en de versie met typefouten te maximaliseren. Hoewel onze methode de robuustheid vergroot, gaat het *contrastive loss*—dat wordt toegepast—uit van een enkele positieve sample en een set negatieve samples per *anchor*, waardoor de meerdere positieve samples (bijv. meerdere synthetische zoekopdrachten met typefouten) niet volledig worden benut. Dit brengt ons bij onze tweede studie, waarin we voorstellen om bestaande typefout-robuuste *dense retrievers* te verbeteren door *contrastive learning* toe te passen dat rekening houdt met meerdere positieve en negatieve samples. Naast het leren van betere representaties voor de ruisgevoelige tekst, kan onzekerheidsschatting ook worden gebruikt om de robuustheid te vergroten. Het *multivariate representation learning (MRL) framework*, voorgesteld door Zamani and Bendersky [187], is de eerste methode die zich richt op het modelleren van onzekerheid in *dense retrieval*. Daarom onderzoeken we in onze derde en laatste studie of het MRL-framework onzekerheid kan vastleggen in zoekopdrachten die typefouten bevatten.

In het tweede deel van deze scriptie richten we ons op een ander onderzoeksthema en bestuderen we hoe we de robuustheid van spraakgestuurde zoekopdrachten met *neural retrieval* kunnen verbeteren. Hier worden de zoekopdrachten uitgesproken, terwijl de documenten in tekstuele vorm zijn. Onze eerste studie richt zich op automatische spraakherkenning (ASR) en ASR-Retriever-pijplijnen, waarbij de gesproken zoekopdrachten worden getranscribeerd met een ASR-model en vervolgens door de *text retriever* worden gehaald. Specifiek willen we de robuustheid van *dense retrievers* beoordelen voor zoekopdrachten die transcriptiefouten bevatten. We tonen aan dat de huidige state-of-the-art *dense retrievers* niet inherent robuust zijn bij zoekopdrachten

met ASR-fouten. Om dit aan te pakken stellen we voor om te trainen met transcripties van synthetisch gegenereerde gesproken zoekopdrachten om de robuustheid te vergroten. Twee belangrijke tekortkomingen van ASR-Retriever-pijplijnen zijn dat de ASR fouten doorgeeft aan de downstream-retriever en dat het trainen van een effectief ASR-model een grote hoeveelheid geannoteerde spraak vereist. Dit brengt ons bij onze tweede studie, waarin we onderzoeken hoe we dergelijke beperkingen van de ASR-Retriever-pijplijnen kunnen oplossen. We stellen een *multimodal*, ASR-vrije *dense retriever* voor die rechtstreeks op de gesproken zoekopdracht kan werken.

In het derde deel van deze scriptie richten we ons op complexe zoekopdrachten (ook bekend als *multi-hop* zoekopdrachten) die niet met één enkel document kunnen worden opgelost. Dergelijke zoekopdrachten vereisen *multi-hop retrieval* om meerdere documenten te verzamelen die gezamenlijk voldoende bewijs kunnen leveren. Specifiek bestuderen we de effectiviteit van *dense retrievers* voor *multi-hop retrieval* met beperkte computerbronnen. We laten zien hoe de prestaties afnemen bij beperkte bronnen en stellen een computationeel efficiënte *hybrid retriever* voor die *lexical* en *dense retrieval* combineert.

In het vierde en laatste deel van deze scriptie richten we ons op een nieuw onderzoeksthema: het verbeteren van domeinaanpassing in kennisgrafen vraag-antwoord (KGVA) met *neural retrieval*. We concentreren ons op een scenario waarin nieuwe domeinen (d.w.z. subgrafen) met nog niet eerder geobeserveerde relaties en entiteiten worden toegevoegd aan de KG—waarbij er tijdens de training geen beschikbare vraag-antwoordparen zijn—en het KGVA-systeem is een pijplijn die bestaat uit het detecteren van entiteitsvermeldingen, het genereren van entiteitskandidaten, relatievoorspelling en antwoordselectie. We richten ons op relatievoorspelling, omdat we hebben vastgesteld dat deze niet goed schaalt naar nieuwe domeinen en we streven ernaar de robuustheid ervan te vergroten voor eerder ongeziene domeinen. We beginnen met het formuleren van relatievoorspelling als een retrievalprobleem waarbij we het tekstuele label gebruiken om de relatie te vertegenwoordigen—in tegenstelling tot de meeste werken die het behandelen als classificatie—waardoor we elke relatie kunnen representeren tijdens de inferentietijd. Bovendien stellen we voor om een *neural retriever* te gebruiken in combinatie met gegevensvergroting om de trainingsset uit te breiden met synthetisch gegenereerde zoekopdrachten voor de nieuwe domeinen.