UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE INFORMÁTICA
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

LEVY DE SOUZA SILVA

Finding Structured Data From Text Using Language Models

Recife
2023

LEVY DE SOUZA SILVA

Finding Structured Data From Text Using Language Models

Tese de Doutorado apresentada ao Programa de Pós-graduação em Ciência da Computação do Centro de Informática da Universidade Federal de Pernambuco como requisito para a obtenção do título de Doutor em Ciência da Computação.

Área de Concentração:
Inteligência Computacional
Orientador:
Luciano de Andrade Barbosa

Recife
2023

Catalogação na fonte
Bibliotecária Nataly Soares Leite Moro, CRB4-1722

S586f Silva, Levy de Souza
Finding structured data from text using language models / Levy de Souza Silva – 2023.
122 f.: il., fig., tab.

Orientador: Luciano de Andrade Barbosa.
Tese (Doutorado) – Universidade Federal de Pernambuco. CIn, Ciência da Computação, Recife, 2023.
Inclui referências e apêndices.

1. Inteligência computacional. 2. Tabelas da internet. 3. Recuperação de tabelas. 4. Correspondência de notícias e tabelas. I. Barbosa, Luciano de Andrade (orientador). II. Título

006.31 CDD (23. ed.) UFPE - CCEN 2024 – 18

LEVY DE SOUZA SILVA

"**Finding Structured Data From Text Using Language Models**"

Tese de Doutorado apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Pernambuco como requisito para a obtenção do título de Doutor em Ciência da Computação.

Aprovado em: 07/12/2023

Orientador: Luciano de Andrade Barbosa

BANCA EXAMINADORA

Prof. Dr. Fernando Maciano de Paula Neto
Centro de Informática - UFPE

Prof. Dr. Leandro Maciel Almeida
Centro de Informática - UFPE

Profa. Dra. Mirella Moura Moro
Departamento de Ciência da Computação - UFMG

Prof. Dr. Rafael Ferreira Leite de Mello
Departamento de Computação - UFRPE

Prof. Dr. Rodrigo Frassetto Nogueira
Departamento de Engenharia de Computação e Automação - UNICAMP

To my father, Jonas Trajano da Silva (in memoriam).

ACKNOWLEDGEMENTS

Primeiramente, gostaria de agradecer a Deus por ter me dado forças para chegar até aqui. O doutorado é uma longa caminhada, e a minha durou cerca de 5 (cinco) anos. 12 (doze) anos atrás, quando eu iniciava minha graduação em computação, eu pensava que o doutorado era o "fim", mas hoje percebo que a vida está apenas começando.

Aconteceu de tudo nesse doutorado, mudança de cidade, de círculos de amizade, e a tão temida pandemia da COVID-19. Comigo não seria diferente. Ao escrever esse texto, a sensação não poderia ser outra: ufa! Que alívio! Dever Cumprido!

Em segundo lugar, gostaria de agradecer aos meus familiares, minha mãe Eneida e meus irmãos Jason, Ayane e Jeise, por todo o apoio durante essa árdua jornada. Que pena meu Pai, Jonas, não está mais aqui para ver isso tudo "terminar". Durante anos da minha vida, eu sempre escutei suas palavras pela boca do meu tio Saulo: "*Meu filho vai ser um doutor!*" Acho ele nem imaginava o que era isso. Meu pai não teve a oportunidade de estudar, não sabia ler nem escrever. Enfim, para os que acreditam, a profecia se cumpriu!

Dando continuidade, gostaria de pontuar um agradecimento especial ao meu orientador Luciano. Luciano é Fod@! Não tenho palavras para descrever a qualidade do profissional que ele é. Luciano é bom em tudo! Literalmente. Com ele eu aprendi várias coisas, desde conselhos para vida até o tradicional roteiro acadêmico. "Levy, tem que ser sangue nos olhos!" Essa frase marcou todo o meu percurso de doutorado.

Em quarto lugar, gostaria de agradecer aos meus colegas de laboratório Neto, Michael e Johny. A pandemia nos separou. Passamos apenas cerca de um ano na labuta diária do laboratório em Recife-PE, onde quase sempre na hora do almoço (nas quentinhas do seu Almir) rolavam àqueles debates filosóficos. Mas, mesmo distantes, vocês foram fundamentais na minha caminhada! Valew Galera! Conseguimos!

Em seguida, não poderia esquecer da minha namorada Ingrid, que me aturava diariamente, por todo esse tempo, ouvindo "delírios" sobre essa tese de computação: *tenho que terminar a tese*. Cuida! Te amo Ingrid!

Por fim, o sexto agradecimento é uma longa história. "Não sei como", mas eu vim parar no Cedro-CE por motivos de trabalho, depois de ter passado por Recife-PE e Lajedo-PE. Assim, gostaria de agradecer também ao meu colega George, por dividir a casa aqui comigo, o famoso "alojamento militar". Ele gostava de computação, apesar de ser formado em história. Logo, eu contava minhas teses computacionais diariamente para ele. Ademais, também gostaria de agradecer aos meus colegas Cledson e Denilson (e o seu *dog simon*) que dividiram a segunda casa aqui comigo e deixaram os meus dias no Cedro mais felizes.

RESUMO

A Internet é uma rica fonte de informação estruturada. De tabelas *Hypertext Markup Language* (HTML) até coleções de dados públicos, existe um enorme conjunto de dados relacionais online. Estudos anteriores estimam que mais de 418 milhões de tabelas, em formato HTML, podem ser encontradas na Internet. Não se limitando a estas, um grande número de repositórios de dados fornecem acesso a milhares de coleções estruturadas. Como resultado, nos últimos anos, vários estudos exploram estes dados em diversas aplicações. Por exemplo, tabelas HTML são geralmente utilizadas na tarefa de perguntas e respostas: considerando uma pergunta e uma coleção de tabelas, o objetivo é encontrar uma tabela, desta coleção, que possa ser utilizada como resposta para esta pergunta. No contexto de dados públicos, a principal aplicação é a busca por conjunto de dados, que encontra uma coleção de dados para um usuário final. O ponto de intersecção destas tarefas é a correspondência de dados estruturados e não estruturados, além de uma tarefa de classificação. Ademais, o principal desafio é construir um modelo computacional robusto para calcular a similaridade entre perguntas e tabelas. Nesse contexto, este trabalho de tese está dividido em três partes. Na primeira, exploramos o problema de recuperação de tabelas para perguntas e respostas, sumarizando as melhores soluções para esta tarefa. Em seguida, introduzimos uma nova tarefa para correlação de notícias e tabelas, aplicadas para expandir o conteúdo das notícias. Por fim, focamos na tarefa de busca por conjuntos de dados. Especificamente, as principais contribuições desta tese são: (I) nós apresentamos uma nova taxonomia para a tarefa de recuperação de tabelas que classifica os métodos de recuperação de tabelas em cinco grupos, desde abordagens probabilísticas até redes neurais sofisticadas. Este estudo também aponta que os melhores resultados para esta tarefa são alcançados por meio de modelos de redes neurais profundas, utilizando redes recorrentes e arquiteturas convolucionais; (II) nós introduzimos um novo modelo de atenção baseado em *Bidirectional Encoder Representations from Transformers* (BERT) para calcular o grau de similaridade entre notícias e tabelas, além de comparar seu desempenho com técnicas de recuperação de informação, codificadores de sentenças e documentos, modelos de correspondência de textos e abordagens de redes neurais. Em resumo, um teste de hipótese confirma que nossa abordagem supera todos os outros modelos considerando uma métrica de classificação média; e (III) nós propomos *Data Augmentation Pipeline for Dataset Retrieval* (DAPDR), uma solução que usa modelos de linguagens para criar perguntas sintéticas para coleções de dados, que são aplicadas no treinamento de modelos supervisionados. Por fim, DAPDR é avaliado utilizando dados experimentais para esta tarefa e modelos densos de recuperação de informação, cujos principais resultados mostram que os modelos ajustados em DAPDR superam estatisticamente os modelos originais em diferentes níveis de *Normalized Discounted Cumulative Gain* (NDCG).

Palavras-chave: tabelas da internet; recuperação de tabelas; correspondência de notícias e tabelas; busca por conjunto de dados; geração de consultas; modelos de linguagem.

ABSTRACT

The Internet is a rich source of structured information. From Web Tables to public datasets, there exists a huge corpus of relational data online. Previous studies estimate that over 418M tables, in Hypertext Markup Language (HTML) format, can be found on the Web. Not limited to them, a large number of data repositories also provide access to thousands of datasets. As a result of that, over the last years, a growing body of work has begun to explore this data for several downstream applications. For example, Web Tables have been widely utilized for the task of Question Answering (QA), whose goal is to retrieve a table that answers a query from a table collection. In the context of datasets, their most popular application is the dataset retrieval task, which aims to find structured datasets for an end-user. The point of intersection for table/dataset retrieval is that they need to match unstructured queries and relational data, in addition to being a ranking task. Moreover, the core challenge of this task is how to construct a robust matching model for computing this similarity degree. Towards this front, this thesis work is divided into three parts. In the first one, we explore the problem of QA Table Retrieval, in which our goal is to outline the best solutions for this task. In sequence, we focus on an unexplored news-table matching problem, whose Web Tables are applied to augmenting news stories. Lastly, we concentrate on the dataset retrieval task. Specifically, we summarize our main contributions as follows: (I) we present a novel taxonomy for table retrieval that classifies the table retrieval methods into five groups, from probabilistic approaches to sophisticated neural networks. Our research also points out that the best results for this task are achieved by using deep neural models, built on top of recurrent networks and convolutional architectures; (II) we introduce a novel attention model based on *Bidirectional Encoder Representations from Transformers* (BERT) for computing the similarity degree between news stories and Web Tables, in addition to comparing its performance against Information Retrieval (IR) techniques, document/sentence encoders, text-matching models, and neural IR approaches. In short, a hypothesis test confirms that our approach outperforms all baselines in terms of the Mean Reciprocal Ranking metric; and (III) we propose Data Augmentation Pipeline for Dataset Retrieval (DAPDR), a solution that leverages Large Language Models (LLMs) to create synthetic questions for dataset descriptions, which are then applied to training supervised retrievers. Finally, we evaluate DAPDR on dataset search benchmarks using a set of dense retrievers, whose main results show that the retrievers tuned in DAPDR statistically outperform the original models at different Normalized Discounted Cumulative Gain (NDCG) levels.

Keywords: web tables; table retrieval; news table matching; dataset retrieval; query generation; large language models.

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

ACORDAR	A Test Collection for Ad Hoc Content-Based (RDF) Dataset Retrieval
ACSDb	Attribute Correlation Statistics Database
ALBERT	A Lite BERT
API	Application Programming Interface
ARCI	Architecture-I
ARCII	Architecture-II
BERT	Bidirectional Encoder Representations from Transformers
Bi-GRU	Bidirectional Gated Recurrent Unit
Bi-LSTM	Bidirectional Long Short-Term Memory
BM25	Best Matching
CBOW	Common Bag Of Words
CKAN	Comprehensive Knowledge Archive Network
CLS	Classification
CLSM	Convolutional Latent Semantic Model
CNN	Convolutional Neural Network
CONV-KNRM	Convolutional Kernel Based Neural Ranking Model
CSV	Comma-Separated Values File
DAG	Directed Acyclic Graph
DAPDR	Data Augmentation Pipeline for Dataset Retrieval
DCG	Discounted Cumulative Gain
DistilBERT	Distilled Version of BERT
DLB	Document Level Boosting
DNN	Deep Neural Network
DPR	Dense Passage Retrieval
DRMM	Deep Relevance Matching Model
DSSM	Deep Structured Semantic Models
ES	Elastic Search
FFN	Feed-Forward Network
FN	False Negative

GMU	Gated Multimodal Unit
GPT	Generative Pre-Training Transformer
GRU	Gated Recurrent Unit
HTML	Hypertext Markup Language
IDCG	Ideal Discounted Cumulative Gain
IDF	Inverse Document Frequency
InPars	Inquisitive Parrots for Search
IR	Information Retrieval
JSON	JavaScript Object Notation
KB	Knowledge Base
KNRM	Kernel Based Neural Ranking Model
LLM	Large Language Model
LSTM	Long Short Term Memory
MAP	Mean Average Precision
MF	Multi-Field
MLP	Multi-Layer Perceptron
MPNET	Masked and Permuted Language Modeling
MRR	Mean Reciprocal Rank
MSMARCO	Microsoft Machine Reading Comprehension
NASA	National Aeronautics and Space Administration
NDCG	Normalized Discounted Cumulative Gain
NLP	Natural Language Processing
NOS	National Ocean Service
NTCIR	National Institute of Informatics Testbeds and Community for Information Access Research
PMI	Point-wise Mutual Information
PROMPTAGATOR	Promptbase Query Generation for Retriever
QA	Question Answering
QUDT	Quantity, Unit, Dimension and Type
RBF	Radial Basis Function
RDF	Resource Description Framework
RNN	Recurrent Neural Network
SBERT	Sentence-BERT

SEP	Separator
SF	Single-Field
SMT	Statistical Machine Translation
TF	Term Frequency
TF-IDF	Term Frequency – Inverse Document Frequency
TLB	Table Level Boosting
TP	True Positive
TREC	Text REtrieval Conference
TTF-ITTF	Table Term Frequency - Inverse Table Term Frequency
TV	Television
USE	Universal Sentence Encoder
XML	Extensible Markup Language

CONTENTS

1 INTRODUCTION

This chapter outlines this thesis work. In this study, we focus on Information Retrieval (IR) tasks for matching unstructured text and structured/semi-structured data. Specifically, we target at the following tasks: (I) Web Table Retrieval; (II) News-Table Matching; and (III) Dataset Retrieval. The point of intersection for such tasks is that they need to match relational data and unstructured information, in addition to being a ranking task.

At this direction, we begin this chapter by introducing the domain of Web Tables, which has been used as motivation for our study (Section 1.1). In sequence, we focus on an unexplored news-table matching problem (Section 1.1.1), whose Web Tables are utilized for news augmentation. Then, we address the dataset retrieval task (Section 1.1.2).

Finally, we formalize each ranking problem we address in this thesis work (Section 1.2), and the research questions, objectives, contributions and main results of our study are also introduced (from Section 1.3 to Section 1.5). We conclude this chapter by presenting the organization of this document (Section 1.6).

1.1 WEB TABLES: A VALUABLE SNAPSHOT OF DATA FROM THE WEB

HTML Tables, a.k.a. Web Tables, are a huge and rich corpus of relational data from the Internet (SUN et al., 2019). In addition to representing complex data, they also enable a quick understanding of entity relationships due to their well-organized structure. In this thesis, we consider Web Tables as a set of relational HTML Tables (i.e., the *<table>* tag from HTML), in which a single relational table contains relation instances and their associated metadata in the form of column headers (BHAGAVATULA; NORASET; DOWNEY, 2015). Based on that, previous studies estimate that over 418 million tables, in HTML format, can be found on the Web (EGGERT et al., 2023). From this total, many tables are used to define page layouts, i.e., the organization of visual elements on a web page (e.g., its buttons, fields or menus). However, a vast number of them contains high-quality structured data about real-world entities and particular categories (CAFARELLA et al., 2008c; MARZOCCHI et al., 2022). For example, Wikipedia[1] alone includes a collection of nearly 2.8M relational tables,[2] which mostly describe entities and their attributes (MARZOCCHI et al., 2022). In summary, Web Tables are a valuable tool to categorize and publish real-world information on the Internet (SHRAGA et al., 2020c).

As a result, over the last years, a growing body of work has begun to explore this table corpus for several downstream applications (BALAKRISHNAN et al., 2015; FEDOROV; MIRONOV; CHERNISHEV, 2023). We assume a table corpus to be a set of structured data composed of tables in HTML format, which are typically extracted from HTML pages

[1] https://en.wikipedia.org/wiki/Main_Page
[2] Extracted from over 21M Wikipedia pages

Figure 1 – An example of the query-table match for a Question-Answering task. We suppose that a user searches for Netherlands Towns, and the relational table lists a sample of the largest cities of this country.

query: major cities of netherlands

Name	Population
Amsterdam , North Holland	741,636
Rotterdam , South Holland	598,199
The Hague , South Holland	474,292

Source: Adapted from Sun et al. (2019)

using the <*table*> tag (CAFARELLA et al., 2008a). Based on that, such tables have been widely utilized for the task of Question Answering (QA), where the goal is to retrieve a table that answers a query from a table corpus (CHAKRABARTI et al., 2020; SUN et al., 2019). QA is a specialized area in the field of IR, in which QA systems provide relevant answers in response to a query proposed in natural language (ALLAM; HAGGAG, 2012). Towards this front, table retrieval for QA is the most popular application for consuming Web Tables, and the core challenge is how to construct a robust matching model for computing the similarity degree between text queries and structured tables. We illustrate a concrete sample of this query/table match in Figure 1, in which we suppose that a user searches for Netherlands Towns. In this example, the content inside the table ⟨*Largest cities of the Netherlands*⟩[3] can be used as an answer to the query ⟨*Major Cities of the Netherlands*⟩ because it contains the response for the user's search intentions. Indeed, some tables provide relevant results for specific queries and, as a result, search engines have also applied them to respond to intent-queries. For example, by searching for the query *List of Brazilian Cities by Population*,[4] Google returns a co-related Wikipedia table in the first ranking position which answers the query.[5]

Not limited to table retrieval, a myriad of work has likewise used Web Tables in further domains. For instance, Web Tables have also been applied to the task of Knowledge Base (KB) Augmentation, in which the goal is to leverage tabular data for increasing KBs (ZHANG; BALOG, 2020). Furthermore, the study *Ten Years of Web Tables* (CAFARELLA et al., 2018) has also discussed a flurry of tasks built around these tables including but not restricted to Table Augmentation, which leverages additional data for expanding an existing table corpus (ZHANG; BALOG, 2017), Table Interpretation, which

[3] https://en.wikipedia.org/wiki/Template:Largest_cities_of_the_Netherlands
[4] Search date: December 11, 2023
[5] https://en.wikipedia.org/wiki/List_of_cities_in_Brazil_by_population

Figure 2 – Improving news understanding by matching a correlated Web Table. A concrete sample of the news/table matching for data augmentation. In this example, we match a news story about rare Da Vinci paintings with a Wikipedia table that lists the most expensive paintings in the world.

Rare Da Vinci painting smashes world records with $450 million sale

Updated 16th November 2017

Leonardo da Vinci's "Salvator Mundi" has become the most expensive artwork to ever sell at auction, going for $450.3 million at Christie's in New York. Dating back to around 1500, the rare painting is one of fewer than 20 authenticated works by the Italian in existence. Original estimates had predicted bids of over $100 million for the piece. But the new record was set after approximately 20 minutes of telephone bidding, far surpassing the previous auction record held by Picasso's "Les Femmes d'Alger," which sold for $179.4 million in 2015.

(a) CNN News

List of most expensive paintings

The most famous paintings, especially old master works done before 1803, are generally owned or held at museums, for viewing by patrons. This list is ordered by consumer price index inflation-adjusted value (in bold) in millions of United States dollars in 2021.

Price	Painting	Artist	Year	Date of Sale
$450M	**Salvator Mundi**	**Leonardo da Vinci**	**1500**	**November 2017**
$300M	**Interchange**	**Willem de Kooning**	**1955**	**September 2015**
$250M	**The Card Players**	**Paul Cézanne**	**1892**	**April 2011**
$210M	**Nafea Faa Ipoipo**	**Paul Gauguin**	**1892**	**September 2014**
$200M	**Number 17A**	**Jackson Pollock**	**1948**	**September 2015**

(b) Wikipedia Table

Source: Adapted from the original web pages

aims to extract semantic knowledge from the table collection, making the tabular data processable by machines (ZHANG; BALOG, 2020), and Table Annotation, which focuses on annotating tables by using an external source (LIMAYE; SARAWAGI; CHAKRABARTI, 2010). For example, in an entity linking domain, the table text can be mapped to a real-world entity by utilizing a KB (BHAGAVATULA; NORASET; DOWNEY, 2015). In summary, Web Tables can be effectively utilized for many downstream scenarios, since they are useful for collecting and organizing real-world information on the Web.

1.1.1 News-Table Matching

With the popularity of Web Tables, several studies have explored this corpus for many specific scenarios (ZHANG; BALOG, 2020). In this context, an unexplored application is to use Web Tables for data augmentation in the news-table matching task, whose tabular data is applied to expanding the content of a news story. Therefore, one of our contributions in this thesis work is a solution towards this goal.

Nowadays, digital news has gained popularity (AKASH et al., 2023; GU et al., 2020). News reading habits have progressively moved from conventional media such as newspapers or Television (TV) to the Internet, where millions of articles are published every day (ATRI; GOYAL; CHAKRABORTY, 2023; SANTOSH; SAHA; GANGULY, 2020). However, given today's news deluge, online readers can be overwhelmed to fully understanding the content of a news story (LEES et al., 2021). One approach to cover this gap is to outline the news by highlighting the most important facts. For example, recent studies sum up news articles by adopting representative headlines for their text (CAI et al., 2023; GAVRILOV; KALAIDIN;

MALYKH, 2019; GU et al., 2020; OMIDVAR; AN, 2023). Likewise, other papers also utilize sentence summarization strategies for creating text document representations (AGARWAL; SINGH; MEEL, 2018; ATRI; GOYAL; CHAKRABORTY, 2023; NALLAPATI et al., 2016; RUSH; CHOPRA; WESTON, 2015). In resume, for both fronts of text summary methods, the goal is just to capture relevant data of the text story.

In this thesis, we argue news understanding can also be enhanced by uncovering contextual data relevant to the article, as structured Web Tables. For instance, popular services such as Google News[6] or Microsoft News[7] could benefit from this *News-Table Matching* by providing associated content to the news articles for their readers. Specifically, we aim to automatically find tables related to news articles. Figure 2 shows a concrete example of this news/table match: Figure 2(a)[8] presents an article about a rare world painting and Figure 2(b)[9] depicts a Wikipedia table that lists the most expensive arts in the world. In this example, the table provides additional information about the central topic of the story, i.e, rare Da Vinci painting. Moreover, by looking at the table, a reader could answer potential questions related to the topic, e.g., what is the second most expensive painting in the world? By looking at the table, the answer is Interchange by Willem de Kooning. Lastly, we can also confirm the selling price of this art by connecting the news and the table ($450 million for these two sources).

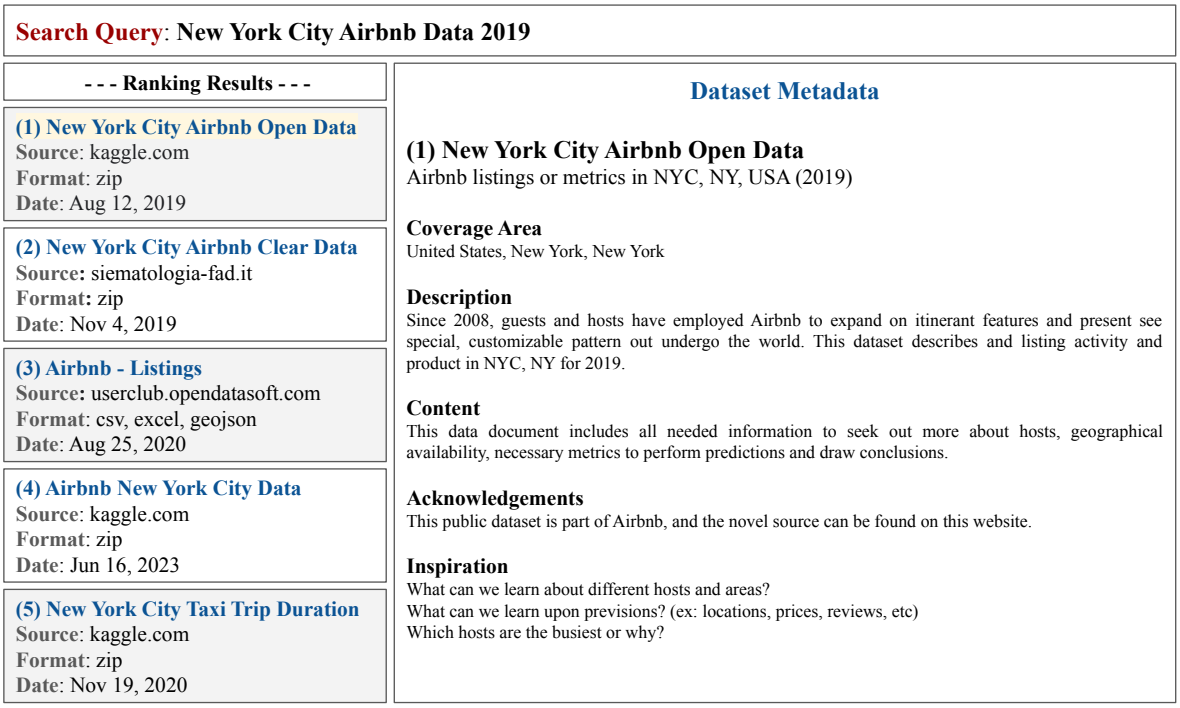
Furthermore, from a fake news perspective, this linking can also improve the credibility of articles and help in preventing rumor spread, since we can verify their facts across two different sources of information. For example, by matching the table in Figure 1(b) with the related news article, *Most Expensive Paintings: A Look at the World's Most Valuable Paintings*,[10] we can verify that they diverge with respect to the price of the *Nafea Faa Ipoipo? painting by Paul Gauguin*, since the article informs that it was sold for around $259 million, while its value in the Wikipedia table is $210 million.

In fact, similar research has shown the news consumption experience enhancement by linking sentences in the article with table cells (KIM et al., 2018). In addition, people can achieve higher recall by jointly reading text and tables than by consuming text alone (GOVINDARAJU; ZHANG; RÉ, 2013). Lastly, web traffic from recent studies has demonstrated that online readers also explore tables inside Wikipedia pages after looking at news articles (LEES et al., 2021).

Towards this front, the task of matching news articles and Web Tables is quite similar to table retrieval for Question Answering. Indeed, its core challenge is also to construct an adapted News-Table matching model for computing this similarity degree. However, this task brings novel challenges to the domain of table retrieval. For example, news stories

[6] https://news.google.com
[7] https://news.microsoft.com
[8] https://edition.cnn.com/style/article/da-vinci-salvator-mundi-sale-christies
[9] https://en.wikipedia.org/wiki/List_of_most_expensive_paintings
[10] https://artincontext.org/most-expensive-paintings

Figure 3 – A snapshot of the Google Dataset Search Framework. We suppose that an end-user searches for the query: *New York City Airbnb Data 2019*, and the engine finds the most correlated datasets for what the user needs (left side of the image). Note that the framework also includes the dataset metadata for the selected dataset in the ranking (yellow mark).



Source: Adapted from Google Dataset Search Framework

can include different entities, categories and objects in the same article. Furthermore, articles are a mixture of unstructured text represented by several aspects, e.g., title, full content, main passage, keywords and so on. In contrast, table retrieval for Question Answering focus on specific intent queries, usually single queries defined by a sequence of few words (SUN et al., 2019; ZHANG; BALOG, 2018), which limits the application of previous solutions for QA table retrieval to our problem since they need to handle distinct news features at the same time for news-table matching.

Given this gap, in this thesis work, we investigate the development of neural-based approaches for the news-table matching task. In addition, we also evaluate the performance of adapted methodologies for QA table retrieval in the context of this task.

1.1.2 Dataset Retrieval

Similar to the problem of table retrieval for QA, dataset retrieval is the task of retrieving structured datasets to an end-user, in which a dataset is a collection of tabular data classified for a particular demand (CHAPMAN et al., 2020). Indeed, both tasks use relational tables to represent data, and one difference is that Web Tables are mostly entity focused, while datasets contain a lot of statistical information (CHEN et al., 2020a). Moreover, Web

Tables and datasets also include contextual data around their corpus, i.e., their table metadata. As a result, retrieval methods designed for table retrieval can be adapted for the dataset retrieval task.

We illustrate an instance of this task by using Google Dataset Search framework[11], as shown in Figure 3. In this example, we assume that an end-user needs to discover Airbnb data for the year 2019 in New York City, and the framework finds the most correlated datasets for this query, in addition to including the dataset metadata for each rank result. Note that queries for dataset search also require the granularity of the data (as e.g., for the year 2019), in addition to focusing on geospatial attributes like cities, regions or countries (as e.g., for the New York data), which differs from a table retrieval task that mostly addresses question-answering queries as, e.g., *What are the Major Cities of Netherlands?* (KACPRZAK et al., 2019).

Finding suitable datasets or tables is a key task for data analysis or exploration. Google Dataset Search have tried to facilitate this data discovery process. By indexing over 30M of structured data, it delivers search tools that let users find datasets on the Web, also providing easier access to the data and its provenance (BENJELLOUN; CHEN; NOY, 2020). Comprehensive Knowledge Archive Network (CKAN)[12] is another well-known backend for managing and sharing open-government data, which makes this data accessible and usable for diverse applications. For example, developers need data to train, validate or enhance machine learning algorithms. Besides, data is also applied to create public policies or identify customer needs (CHAPMAN et al., 2020). Not limited to such tools, Auctus[13] is a different search engine for data discovery and augmentation, in which users can also explore search results through dataset descriptions, geographical information and summary statistics (CASTELO et al., 2021).

With the popularity of the Google Dataset Search framework, the task of searching for datasets has received more attention (CHEN et al., 2020a). Indeed, a shared event for dataset retrieval at the NTCIR-16 conference has attracted researchers from all over the world (KATO et al., 2022). In short, its core tasks include dataset retrieval, QA for structured datasets and data discovery interfaces.[14] Not limited to that, Chapman et al. (2020) also survey a list of open problems for this task including but not restricted to: (I) the development of search paradigms for improving ranking results; (II) the creation of alternative types of queries for specialized accesses (i.e., moving beyond keywords queries in the search); and (III) the construction of intuitive search interfaces for user navigation and exploration (CHAPMAN et al., 2020). Also, other studies have focused on the task of Query Comprehension, whose goal is to analyze the structure of queries issued on this open data portals (KACPRZAK et al., 2017; KACPRZAK et al., 2018; KACPRZAK et al., 2019).

[11] https://datasetsearch.research.google.com
[12] https://ckan.org/government
[13] https://auctus.vida-nyu.org
[14] https://ntcir.datasearch.jp

In addition, there is, however, a lack of available labeled queries mapped to datasets for this task. For example, in NTCIR benchmark, a public corpus for this task derived from open-data portals, just around 200 questions are available (KATO et al., 2021).[15] ACORDAR is another popular test set for dataset retrieval, but it has only 493 ground truth queries (LIN et al., 2022a). Given this gap, developing accurate retrieval models for dataset search becomes challenging since the number of annotated queries is very scarce in each benchmark. In addition, both collections have used human annotators or crowd-sourcing workers to produce questions for datasets, which demands manual efforts and relevance supervision.

One approach to cover this gap is to utilize Large Language Models (LLMs) for text query augmentation. In fact, text data augmentation has emerged as a practical strategy for many Natural Language Processing (NLP) tasks. Inspired by the popularity of LLMs, many studies have used them for producing task-specific training data (SAAD-FALCON et al., 2023). For example, Dai et al. (2022) use LLMs as a few-shot query generator, and retrievers are tuned on this data for improving retrieval accuracy. At the same direction, Bonifacio et al. (2022) create matching queries for text documents, later used for training supervised models. Not limited to that, Nogueira et al. (2019) also target LLMs for data augmentation, in which synthetic queries are used for document expansion. Such approaches are promising since labeled data is limited to some tasks.

Based on that, in this thesis work, we also evaluate the application of LLMs for producing synthetic queries in the dataset retrieval task. The query-dataset pairs are then used for fine-tuning dense retrieval approaches at the target task for ranking.

1.2 PROBLEM STATEMENT

In this thesis, we discuss three correlated tasks for structured/semi-structured data retrieval: QA Table Retrieval, News-Table Matching, and Dataset Retrieval. Since each task utilizes relational tables to represent information, we formalize them as follows.

Given a query q_i and a set of structured tables $T = \{t_1, t_2, t_3, ..., t_n\}$, our goal is to find relevant tables t_i to q_i. Note that the notion of relevance is very broad, since web tables or structured datasets can summarize the search or bring contextual data to the query. In addition, for each task, we suppose that the structured tables are formed by a set of lines and rows (similar to a relational database). However, as tables/datasets also include contextual information around their data (as e.g., their title or short description), we further assume them as a semi-structured corpus.

Specifically for the queries, in the context of news-table matching task, we consider that they are composed of the news aspects, such as its title or short description (i.e., the unstructured information). For QA table retrieval or dataset retrieval, queries are usually

[15] This corpus is part of a shared event for dataset retrieval in NTCIR Conference that has attracted researchers from all over the world (KATO et al., 2022)

defined by a sequence of a few words proposed in natural language. Given that, we also suppose that each task is a ranking problem, i. e., their objective is to learn a scoring function $f : q_i \times t_j \mapsto \mathbb{R}$ that scores the structured data in T for a given query q_i.

1.3 RESEARCH QUESTIONS

This thesis work concentrates on tasks for structured/semi-structured data retrieval. Toward this front, we have discussed the problems of news-table matching and dataset retrieval, in addition to introducing the domain of Web Tables, which has been used as motivation for our study, specifically for the context of table retrieval approaches.

The point of intersection for such tasks is that they need to match unstructured information and structured/semi-structured data, also being a ranking task. As a result, the methods designed for Web Table retrieval can also be adapted to the dataset retrieval task, since they use tabular data to describe real-world information, in addition to containing the surrounding text as their corpus metadata. Besides, we also pointed out that there is a very limited labeled corpus of training data for the dataset retrieval task. In this front, our goal is to investigate the use of LLMs for text data augmentation. Based on that, in this thesis, we investigate the following research questions, which we classify according to each task domain (Web Tables, News-Table Matching and Dataset Retrieval):

- **Web Tables**
  - RQ1: How can we retrieve contextual tables for a query from a table corpus?
  - RQ2: How to calculate the similarity between unstructured queries and semi-structured/structured Web Tables?
  - RQ3: Which table aspects such as headers, caption or body are more relevant to the match in the context of table retrieval?
- **News-Table Matching**
  - RQ4: How to compute the matching degree between News Stories and Web Tables for the news-table matching task?
- **Dataset Retrieval**
  - RQ5: Can we improve the efficacy of supervised retrievers by fine-tuning them on LLMs augmented queries for the dataset retrieval task?

1.4 OBJECTIVES

This thesis work has the following goals: (I) propose a novel taxonomy for the task of QA table retrieval; (II) develop an end-to-end solution for the news-table matching task; and

(III) create a new pipeline for the dataset retrieval task. Towards this front, we list the following specific objectives:

- Investigate previous solutions for the task of Web Table retrieval in the literature;
- Propose a novel news-table matching model to compute the similarity degree between unstructured news stories and structured Web Tables;
- Collect a set of news articles and Wikipedia tables to construct an experimental training corpus for the news-table matching task;
- Generate supervised training data for the dataset retrieval task by using LLMs;
- Evaluate the performance of IR methods, sentence encoders, neural matching models and dense retrievers in the domains of table/dataset retrieval;
- Compare our solutions against the literature baselines.

1.5 CONTRIBUTIONS

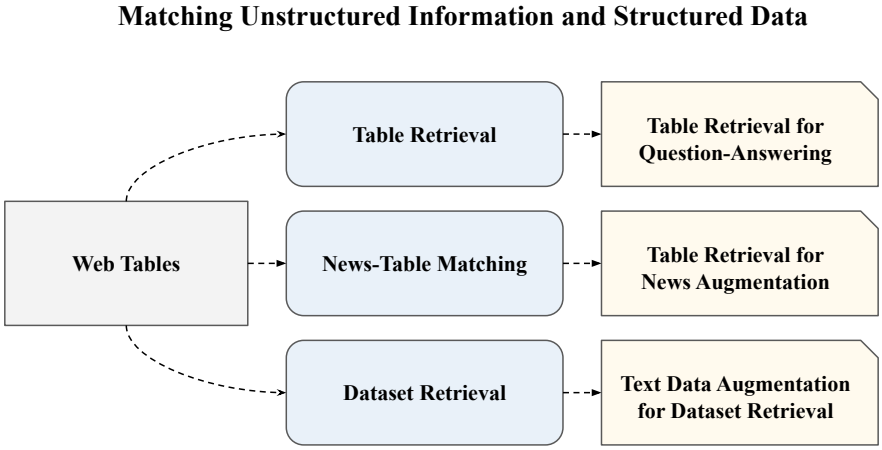
This thesis work addresses IR tasks for structured data retrieval, which we summarize in Figure 4. By assuming the domain of Web Tables, we target at the tasks of Table Retrieval, News-Table Matching, and Dataset Retrieval. For the first task, we focus on table retrieval for question-answering. In the other ones, our goals include table retrieval for news augmentation, and text data augmentation for dataset retrieval. In addition, since news-table matching and dataset retrieval are novel matching problems, in which the number of related work is very scarce, we begin by surveying QA table retrieval approaches from literature, in order to adapt their best methodologies for our tasks, as both of our domains use tabular data to represent information, similar to QA table retrieval. Besides, in the context of dataset retrieval, there also is a very limited labeled matching corpus, which limits the development of supervised approaches to such task. We cover this gap by using LLMs for text data augmentation.

At this direction, in order to answer the research questions (from RQ1 to RQ5), we classify our thesis contributions into the three following parts.

Part I - Web Tables. The first part of our thesis focuses on the questions RQ1, RQ2 and RQ3, and our contribution is an extensive survey on table retrieval solutions for question-answering based on 16 core studies. For that, we consider publications up to the year 2020, in addition to adopting a snowballing methodology for the literature review, in which our start set is the work presented by Zhang and Balog (2018).[16] Specifically, we propose a new table retrieval taxonomy that classifies the table retrieval methods into five groups: *Probabilistic-Based*, *IR-Based*, *Entity-Based*, *Feature-Based* and *Network-Based*.

[16] This is a relevant study for table retrieval published at a CSRanking conference

Figure 4 – An overview of this thesis scope. By using the domain of Web Tables, we classify our contributions into three parts: Table Retrieval for QA, Table Retrieval for News Augmentation and Text Data Augmentation for Dataset Retrieval.



Source: Created by the author

In our survey, we cover studies for QA table retrieval from probabilistic approaches to novel sophisticated deep learning network architectures.

Furthermore, beyond introducing open research challenges for table retrieval including more evaluation datasets and ablation studies on table-features, we also present a comparative analysis concerning table retrieval methodologies, evaluation benchmarks and common table aspects (we classify them into query dependent and query independent features, as well as document fields and document measures). Our research overall points out that the best results for this task are achieved by using deep neural models, build on the top of recurrent networks, convolutional architectures and gated units, which outperform the traditional information retrieval methods.[17]

Part II - News-Table Matching. The second part of this thesis handles the news-table matching tasks, i.e., our research question RQ4. In the context of this task, this thesis work presents the following contributions. We first introduce the problem of collocating news articles with structured web tables as a novel ranking task, in addition to formalizing the most used matching features for this task. Moreover, since the experimental benchmarks for this task are restricted, we present the first news-table corpus from literature. By crawling Wikipedia pages, we collected 275,352 news articles and 298,792 web tables. In addition, our experimental ground truth contains over 93k news-table matching pairs created by distant supervision strategies (SMIRNOVA; CUDRÉ-MAUROUX, 2019). This corpus is publicly available.[18]

[17] Since our survey dates from 2020, we do not consider transformer models in our investigation. However, such models need to be considered for further analysis in the context of this task, as transformer networks have achieved greater results in different natural language processing tasks.

[18] https://github.com/levysouza/News-Table-Matching

Not limited to that, our core contribution for this task is an end-to-end solution for matching news articles and web tables. Based on that, we propose a novel BERT-based attention model for computing this similarity degree. Our work goes beyond the previous studies that apply BERT for retrieval, in addition to fine-tuning it to the target task (SUN et al., 2019; LEES et al., 2021; NOGUEIRA; CHO, 2019), since we also consider matching information from attention matrices over the inputs. In resume, our solution has two cascaded steps. First, similar to previous work (SHRAGA et al., 2020a; SHRAGA et al., 2020c; SHRAGA et al., 2020b; SUN et al., 2019), we retrieve a set of candidate tables by using a standard Information Retrieval (IR) approach, whose goal is to efficiently find the highest number of relevant tables for the matching model. Next, we use the proposed model to re-rank the candidates in order to obtain the best matching tables.

Another contribution to this task is that we perform an extensive experimental evaluation. Specifically, we compare the performance of our solution with standard IR techniques, document/sentence encoders, text matching models, neural IR approaches and dense retrievers for this task, also assessing both single and multi-field (document) ranking methodologies in the experimental setup. Overall, a statistical hypothesis test confirms our method statistically outperforms all baselines in terms of Mean Reciprocal Ranking (MRR@50). Concerning accuracy, our model achieves near 55% accuracy@1 as opposed to the best baselines varying between 13% and 48%. Such results demonstrate our model re-ranks the best matching table for a news story at the first ranking positions.

Part III - Dataset Retrieval. The third part of this thesis concentrates on using LLMs for Dataset Retrieval, in order to answer the research question RQ5. For this task, the contribution of this thesis is a Data Augmentation Pipeline for Dataset Retrieval (DAPDR), a solution that leverages LLMs to generate synthetic questions for dataset descriptions. These question-description pairs are then employed to fine-tune dense retrievers for dataset ranking. Similar to previous work (BONIFACIO et al., 2022; DAI et al., 2022; JERONYMO et al., 2023), we also focus on LLMs for text data augmentation and assume the query-description pairs as soft-matches to our task. To the best of our knowledge, we are the first study to apply LLMs in the context of dataset retrieval.

In addition to that, another contribution for this task is that we evaluate DAPDR on the dataset search benchmarks using a set of dense retrievers for semantic search. In our experiments, we have assessed the performance of BERT, SBERT, MPNET and DPR,[19] which have shown strong results for tasks like Web Search and Question Answering (GAO et al., 2022; KARPUKHIN et al., 2020; LEE; CHANG; TOUTANOVA, 2019). Our goal is to compare whether the retrievers tuned on the augmented data utilizing DAPDR outperform the original models. For that, we fine-tune the retrievers on DAPDR and compare them with the original models. We also assess the performance of IR methods for comparison purposes. Our results show that the tuned retrievers statistically outperform the original

[19] https://www.sbert.net/docs/pretrained_models.html#

models at different NDCG levels. Their efficacy improves from 6% to 69% for NDCG@5 and, in some cases, three times better than the pre-trained approach. Compared to IR techniques, the models tuned on DAPDR surpass them from over 6% to 34% for NDCG@5.

1.6 DOCUMENT ORGANIZATION

The remainder of this document is organized as follows.

- **Chapter 2 - Background**. This chapter covers the IR background for this thesis work. We formalize the tasks of question-answering and ad-hoc document search, in addition to defining IR methods for text retrieval. Furthermore, we also describe the IR workflow that we consider in this thesis work for our experimental setup. In sequence, we present neural IR models for text representation and for text matching, also introducing dense retrieves for this task. We conclude this chapter by focusing on text data augmentation, in which Large Language Models (LLMs) are utilized for text query generation.

- **Chapter 3 - Table Retrieval**. This chapter formalizes the table retrieval task for the domain of question-answering, in addition to depicting some background about Web tables. Furthermore, we present our table retrieval taxonomy, and representative solutions for this task are also detailed. In sequence, we compare current methodologies, query/tables features and evaluation benchmarks for table retrieval. We conclude this chapter by covering a set of open challenges for this task.

- **Chapter 4 - News-Table Matching**. This chapter introduces the task of table retrieval for news argumentation, in which our goal is to match news articles and Web Tables. Moreover, we further describe our BERT-based attention model for news-table matching, also detailing each block of our neural network. In sequence, we also define our experimental setup for this task. We conclude this chapter by presenting our main retrieval results for the news-table matching task.

- **Chapter 5 - Dataset Retrieval**. This chapter focuses on our data augmentation task for dataset retrieval. Besides, we also present DAPDR (**D**ata **A**ugmentation **P**ipeline for **D**ataset **R**etrieval), a solution that uses LLMs for producing query-dataset samples for fine-tuning dense retrievers at the target task. We conclude this chapter by depicting the experimental setup and our results for dataset retrieval.

- **Chapter 6 - Concluding Remarks**. This chapter summarizes the contributions of our study according to each task domain: Table Retrieval, News-Table Matching and Dataset Retrieval. In addition, the research questions are also answered. Lastly, we further point out limitations and future work of this thesis work.

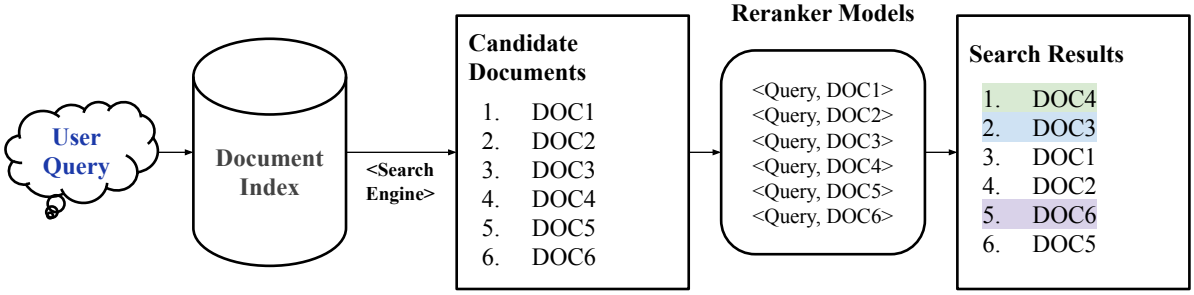
2 BACKGROUND

This chapter covers the IR background in the perspective of our study: text match and text retrieval for structured/semi-structured data. Towards this direction, we start this chapter by defining traditional IR tasks for text retrieval such as Ad-hoc Document Retrieval and Question-Answering (Section 2.1). In addition, we also explain IR methods for this task including TF-IDF and BM25 (Section 2.2). In sequence, we introduce neural IR techniques for text representation (Section 2.3.1), neural IR models for text matching (Section 2.3.2) and dense neural models for text retrieval (Section 2.3.3). Not limited to that, we also point out evaluation measures in IR that we use in this thesis work (Section 2.4). Lastly, we focus on text data augmentation, in which a Large Language Model (LLM) is utilized for synthetic query generation in a target task (Section 2.5), and the concluding remarks of this chapter are also presented (Section 2.6).

2.1 IR TASKS

Information Retrieval is a field of computer science focused on identifying and retrieving resources that are relevant to an information need (such as documents or images). Based on that, the core of an IR system is usually formed by a set of IR components including crawling, indexing, searching, ranking, and so on (SCHÜTZE; MANNING; RAGHAVAN, 2008). In this thesis work, we target IR for text match and text retrieval, which we formalize as follows. Given a text query, usually a sequence of few words proposed in natural language describing real-world entities, particular categories or statistical data, a system needs to return a ranked list of search results according to the relevance for this query

Figure 5 – A two cascade workflow for the IR tasks that we consider in this thesis work. First, a search engine efficiently retrieves a set of candidate documents from the corpus for each query. Then, a sophisticated reranker model extracts relevant matching signals from this subset for re-ranking. Note that *DOC4*, *DOC3* and *DOC1* have shifted their position after the re-ranking step.



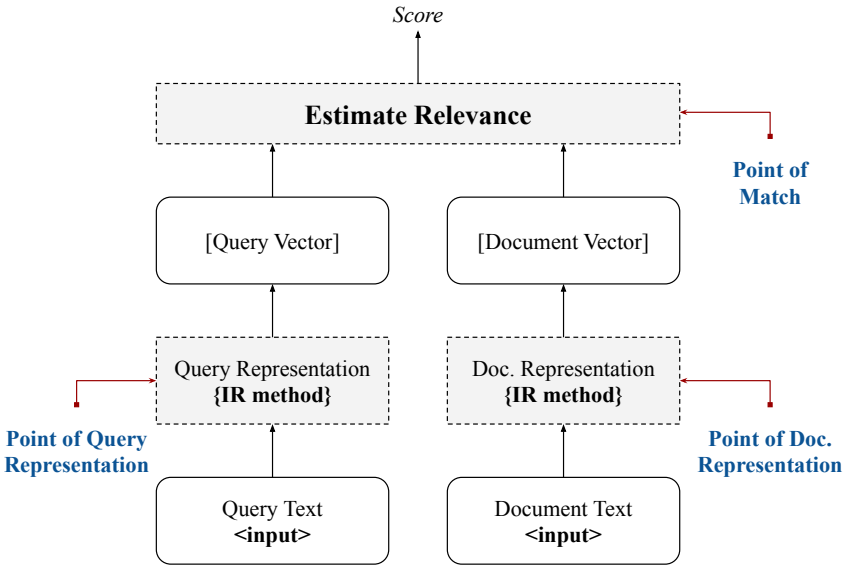
Source: Created by the author

(from highest to smallest based on a similarity metric). Note that the notion of relevancy is very restricted to the task domain, since this ranked list can respond to the query, bring contextual information, or summarize the user's intentions. In addition, the search results can be either long text documents, which stands for the task of *Ad-hoc Document Retrieval*, aiming to retrieve the whole text document, or short text passages from them, i.e., the task of *Question-Answering*, whose goal is to produce a brief answer to a query by combining multiple text passages from distinct sources or by summarizing complete text documents (MITRA; CRASWELL, 2018).

Hence, we also assume the task of document retrieval as a two-cascade workflow, as shown in Figure 5. First, a search engine efficiently finds candidate documents for the query by using a traditional IR methods such as the algorithm BM25, whose goal is to quickly retrieve documents to the query. This subset of the index is then applied to sophisticated reranker models, which aims to extract more suitable matching features from the query and the document terms in order to produce the final ranking. In the scope of our tasks (*news-table matching and dataset retrieval*), we consider that the documents are represented by Web Tables or structure datasets.

Therefore, regardless of the IR task we assume, any text match/retrieval model needs to cover an important set of challenges including but not restricted to: (I) *the semantic understanding*, to consider inexact matching between query and documents or text passages; (II) *the robustness to corpus variance*, to treat those cases where the distri-

Figure 6 – The core of text retrieval/match for IR tasks. Mitra and Craswell formalize this task based on two main points: Query/Document Representation, which produces query/document vectors for the inputs; and Query/Document match, which measures their similarity degree.



Source: Adapted from Mitra and Craswell (2018)

butions of the train/test corpus are different; (III) *the sensitivity to context*, to leverage implicit and explicit context information for each term in the corpus; (IV) *the robustness to rare inputs*, to handle with less frequently queries in the test; and (V) *the efficiency*, to efficiently search for the queries over a larger document index.

In addition, the task of text retrieval or text match generally comprises two basic steps, which we illustrate in Figure 6: *(I) point of representation*; and *(II) point of match*. The first one focuses on query/document text representation, aiming to produce a fixed-size vector for each side of the input. In contrast, the point of match analyzes the correlation between query/document terms by employing a similarity measure. The cosine distance is generally applied for computing this matching degree. As follows, we introduce the IR methods that we consider in this thesis work for text representation and text match.

2.2 TRADITIONAL IR METHODS FOR TEXT RETRIEVAL

There exist a lot of methods for the task of text retrieval including statistical algorithms (SALTON; YANG, 1973), language models (PONTE; CROFT, 1998), translation approaches (BERGER; LAFFERTY, 1999) and dependence strategies (METZLER; CROFT, 2005). Term Frequency – Inverse Document Frequency (TF-IDF) (SALTON; YANG, 1973) and the ranking algorithm BM25 (SALTON; YANG, 1973) are widely used for this task, and they serve as important baselines for experimental comparison (MITRA; CRASWELL, 2018).

TF-IDF is a basic IR method that represents query/documents terms by using a sparse vector based on Term Frequency (TF), which expresses the number of times that a term occurs in a document, and Inverse Document Frequency (IDF), which denotes the frequency of a term for the whole corpus. The IDF score is an adjusted weight for some terms that appear more frequently in the corpus. As a result, their IDFs are lower because they are less representative terms for the document. In contrast, rare terms that provide much information for the document attain higher scores for their IDFs. In short, TF-IDF is an IR methodology to represent queries and documents using sparse vectors, which is usually combined with a similarity metric in the target retrieval task, as cosine distance.

At this direction, for each query or document term in the corpus, its TF-IDF is computed according to Equation 2.1:

$$TFIDF(t, d, D) = TF(t, d) * IDF(t, D) \tag{2.1}$$

where $TF(t, d)$ is the term frequency for a term t in a document d, and $IDF(t, D)$ denotes the inverse document frequency of the term t for the whole corpus D. Note that there are many methods for calculating the TF and the IDF scores, such as the raw count itself, boolean frequencies or logarithmically scaled frequency. Furthermore, in a document retrieval task, documents are usually ranked by computing their cosine distance over the TF-IDF vector for the query, according to Equation 2.2, in which $\hat{q}$ represents the vector

for the query, $\hat{d}$ is the vector for the document, n denotes the dimension of the vector, and the symbol $(\cdot)$ is the dot product between $\hat{q}$ and $\hat{d}$.

$$cos(\hat{q}, \hat{d}) = \frac{\hat{q} \cdot \hat{d}}{||\hat{q}|| * ||\hat{d}||} = \frac{\sum_{i=1}^{n} \hat{q}_i * \hat{d}_i}{\sqrt{\sum_{i=1}^{n} \hat{q}_i^2 * \sum_{i=1}^{n} \hat{d}_i^2}} \tag{2.2}$$

In addition to the TF-IDF representation, another relevant method for text retrieval is the algorithm BM25: a raking algorithm based on the probabilistic relevance framework that uses term-frequency weighting and document length for ranking documents. Given a query Q, containing keywords $q_1, ..., q_n$, the BM25 score of a document D is computed according to Equation 2.3:

$$BM25(Q, D) = \sum_{q_i=1}^{n} IDF(q_i) * \frac{TF(q_i, D) * (k_1 + 1)}{TF(q_i, D) + k_1 * (1 - b + b * \frac{|D|}{avgdl})} \tag{2.3}$$

where $IDF(q_i)$ represents the IDF of the term q_i, $TF(q_i, D)$ is the frequency of the term q_i for a document D, k_1 and b are weight parameters for term-frequency and document-length generally tuned on the validation dataset (as e.g., for the common values $k_1 \in [1.2, 2.0]$ and $b = 0.75$), $|D|$ is the length of the document D in words, and $avgdl$ is the average document length in the text collection. Based on that, there is no need to compute the cosine distance when the BM25 algorithm is used for retrieval, since this score is also utilized to rank documents across the queries. In summary, both methods Cos(TF-IDF) and BM25 are term-overlap IR approaches, which focus on the lexical correlation between the query and the document terms. We also evaluate their ranking performance in the context of our table retrieval tasks.

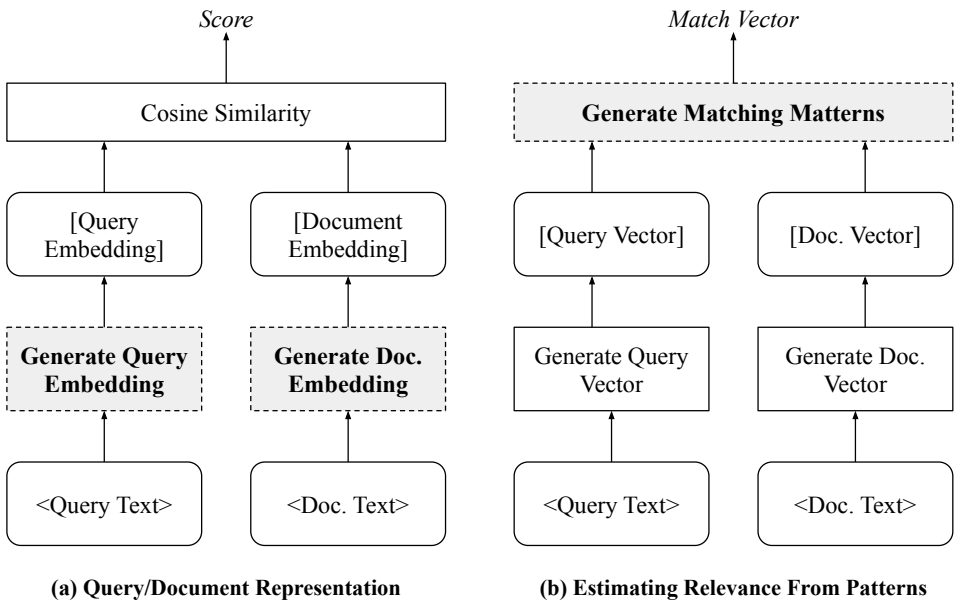
2.3 NEURAL INFORMATION RETRIEVAL

In this section, we cover the background for neural IR approaches, which are built on top of neural network architectures (HAYKIN, 1998) and deep learning models (GOODFELLOW; BENGIO; COURVILLE, 2016).

With the popularity of neural networks, many studies have been utilizing them for text retrieval tasks. According to Mitra and Craswell (2018), neural IR models can be employed for several sub-tasks of text match/retrieval including the task of: (I) learning query/document term representations (HUANG et al., 2013); (II) learning to rank by using manually designed features (LIU, 2009); (III) estimating relevance from patterns of exact matches (GUO et al., 2016); and (IV) augmenting the queries employing neural embeddings (DIAZ; MITRA; CRASWELL, 2016).

In this thesis work, we focus on the neural IR tasks *I* (*query/document representation*) and *III* (*query/document relevancy*) as shown in Figure 7: (a) we use neural models for creating word embedding representations for the text data, in which the cosine similarity estimates the correlation between query and document; and (b) we apply neural models

Figure 7 – Neural IR models for text retrieval and text matching: (a) neural models are applied for learning query and document representation; and (b) neural models are utilized for estimating relevant match patterns from the inputs.



Source: Adapted from Mitra and Craswell (2018)

for getting matching patterns from the input, whose matching vectors can also be used for text match/classification in any machine learning algorithm, i.e., they are employed in the domain of representation learning. As follows, we introduce neural networks for text representation and present a set of neural IR models and dense retrievers for this task.

2.3.1 Neural IR Models for Text Representation

The most popular neural model for text representation is *Word2Vec*: a shallow network for learning word embedding vectors according to Skip-Gram and Common Bag Of Words (CBOW) architectures (MIKOLOV et al., 2013a). In addition to capturing the context of a word in a sequence, they also learn its relation with other words (i.e., their semantic and syntactic similarity). We illustrate their methodologies in Figure 8: (a) CBOW architecture and (b) Skip-Gram architecture.

Both networks include an input/output layer, in which each word is represented by a vector based on an *one-hot-encoder* approach (i.e., words are mapped to binary vectors), and a hidden layer, aiming to represent the neural embeddings (i.e., the word projection). The goal of *CBOW* is to predict a context word $w(t)$ based on its neighbors (i.e., $w(t-2), w(t-1), w(t+1), w(t+2)$), and Skip-Gram infers C context words for $w(t)$. Note that the CBOW architecture also includes a merging function for combining the input words as, e.g., for the sum or average of them. Lastly, in a training task for predicting the

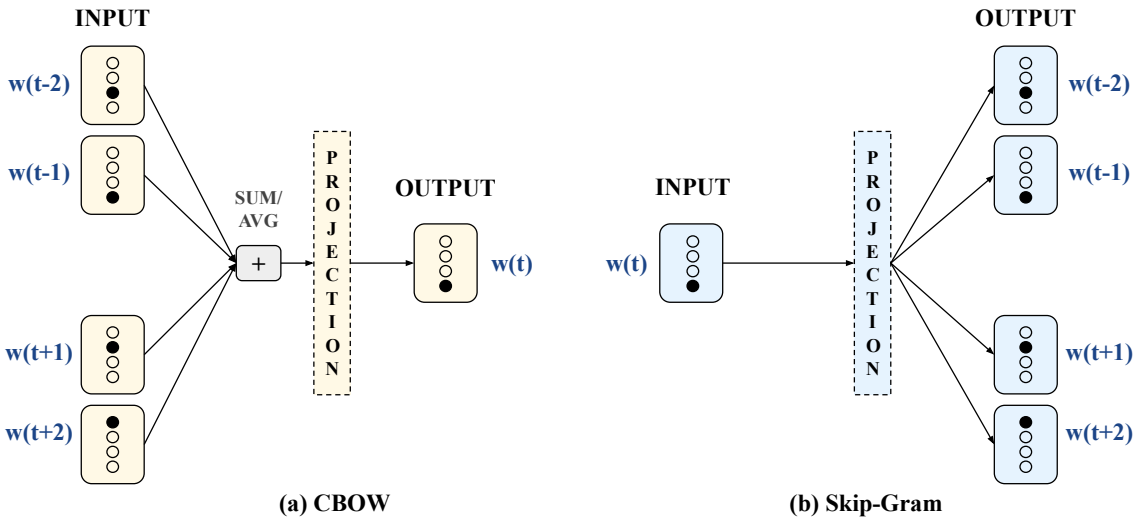
target word, such networks learn its vector representation by using the back-propagation algorithm, in which the hidden layer is the embedding vector to the target (RONG, 2014).

While *Word2Vec* encodes words or tokens to fixed-size dense vectors, other neural IR models focus on embed sentences, paragraphs or full text documents as, e.g., the models *Doc2Vec* (LE; MIKOLOV, 2014), Universal Sentence Encoder (USE) (CER et al., 2018) and Bidirectional Encoder Representations from Transformers (BERT) (DEVLIN et al., 2019). We detail such approaches as follows.

Doc2Vec is similar to *Word2Vec* but it can also be used for representing variable-length pieces of text, such as sentences or entire documents. We illustrate its network architecture in Figure 9. In contrast to *Word2Vec*, its goal is to jointly embed paragraphs and context words in a single neural network architecture for predicting a target word. For that, it uses text paragraphs that are usually chosen by considering a sliding window over the whole text, and the context words sampled from them. Note that *Doc2Vec* maps every part of the input to a unique vector, which are then aggregated to predict the next word in a sequence. In this example, the context words are represented by $<the>$, $<cat>$ *and* $<sat>$, while the target word is the token $<on>$. The key idea for this approach is that the paragraphs can also contribute to the target prediction. After being trained, such vectors can be employed as dense embeddings for sentences or documents.

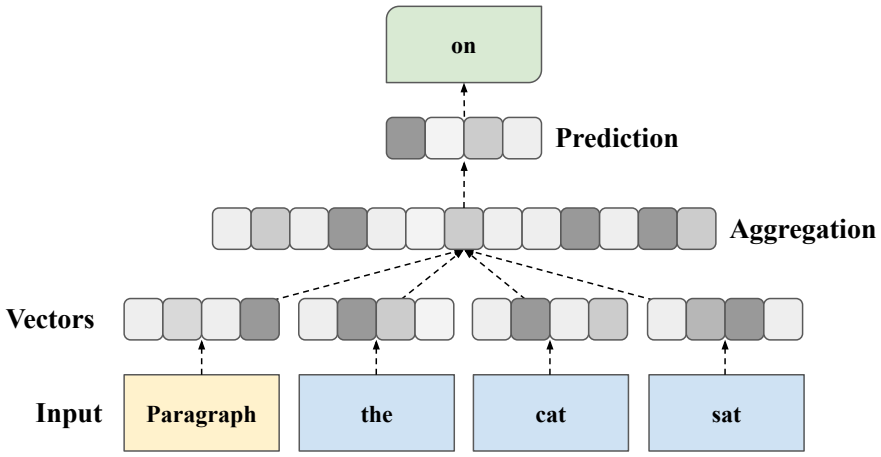
In addition to *Doc2Vec* (LE; MIKOLOV, 2014), other popular models for text representations include *USE* (CER et al., 2018) and *BERT* (DEVLIN et al., 2019). *USE* is an encoder model for general-purpose NLP tasks, which utilizes transformer blocks to create dense embeddings for text input sentences. In this network architecture, the attention mecha-

Figure 8 – The two most popular *Word2Vec* architectures for text representation: (a) CBOW, which predicts a target word $w(t)$ based on its neighbors; and (b) Skip-Gram, which infers C context words for an input $w(t)$.



Source: Adapted from Mikolov et al. (2013a)

Figure 9 – The core of the *Doc2Vec* architecture. Similar to *Word2Vec*, it encodes paragraphs and context words in a unique neural network for predicting the target word, in which the paragraphs are chosen by considering a sliding window over the whole text, and context words are sampled from them.

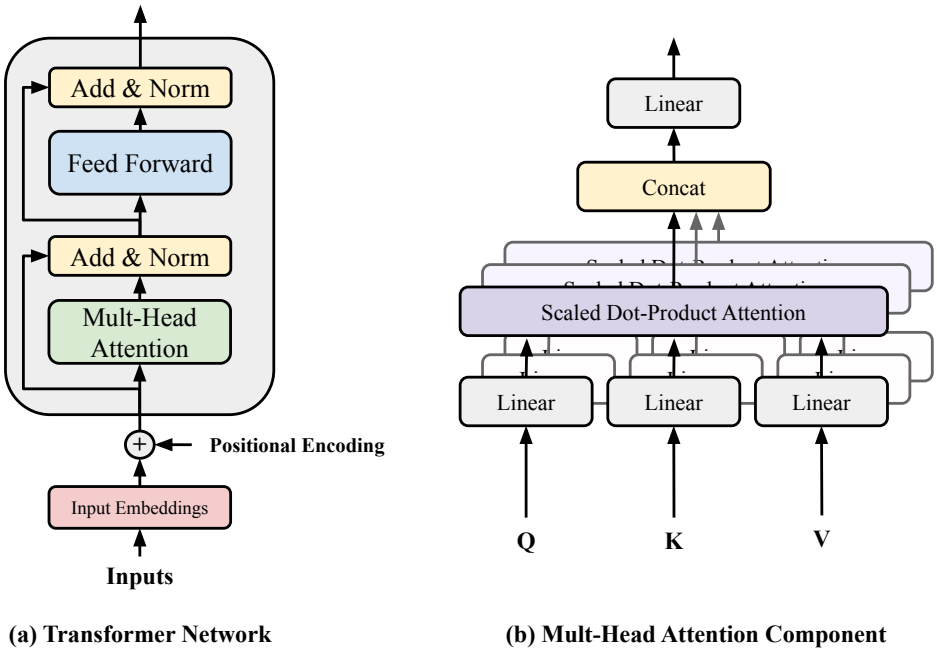


Source: Adapted from Le and Mikolov (2014)

nism is applied to learn the context-aware word representations, which assume both word ordering and word relationship. In addition, each word is mapped to a fixed-length vector by computing the element-wise sum for their context vectors. Based on that, given an input text sentence, USE outputs a 512-dimensional vector as the sentence embedding. This model was originally pre-trained on a set of similarity-related tasks such as Textual Entailment and Question-Answering.

Likewise, *BERT* is another sentence encoder that uses bidirectional transformer networks for language representations, pre-trained on a large corpus on the Web (DEVLIN et al., 2019). A transformer is a sequence-to-sequence model formed by an encoder and a decoder network (VASWANI et al., 2017), in which each one is a stack of L identical blocks. In this architecture, each encoder block is composed of multi-head self-attention components and a position-wise Feed-Forward Network (FFN). Regarding the decoder blocks, they further insert cross-attention approaches between multi-head self-attention and position-wise FFN (LIN et al., 2022b). Based on that, the BERT architecture comprises a set of multi-layer transformer blocks (L), a hidden size (H), and several self-attention heads (A), in which on the BERT-base model $L = 12$, $H = 768$, and $A = 12$. The core idea of BERT is to stack various transformer blocks in a single neural network in sequence. Moreover, it also uses WordPiece embeddings as the token vocabulary (WU et al., 2016), whose last hidden state, the $[CLS]$ vector, stands for the sentence embedding. BERT was originally pre-trained on two distinct tasks: (a) *Masked Language Modeling*, where the goal is to predict the masked token in a sentence; and (b) *Next Sentence Prediction*, whose aims to infer sentence relationships. However, it can easily be adapted for many NLP domains.

Figure 10 – The core of the transformer architecture in two fronts: (a) a transformer block composed of a multi-head attention component, normalization layers and feed-forward networks; and (b) a multi-head attention block based on scaled dot-product or self-attention mechanism.



(a) Transformer Network **(b) Mult-Head Attention Component**

Source: Adapted from Vaswani et al. (2017)

Finally, since BERT and USE are built on top of the transformer network (VASWANI et al., 2017), we illustrate its core architecture in Figure 10: (a) the transformer network and (b) the multi-head attention component. The main idea of the transformer is the multi-head attention component (green block), which contains a set of scaled dot-product attention or self-attention mechanisms over the same input for producing the context-aware sentence representation. Moreover, positional encoding is also included in this network-branch, which maps the word position for the text input. In contrast to Recurrent Neural Networks (RNNs) (ELMAN, 1990), Long Short Term Memorys (LSTMs) (HOCHREITER; SCHMIDHUBER, 1997) and Gated Recurrent Units (GRUs) (CHO et al., 2014a), which only capture token relationships for closely words, attention architectures can also learn long text dependencies in a text sentence. Based on that, the multi-head attention mechanism is calculated according to Equation 4.1:

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_K}})V \tag{2.4}$$

where Q corresponds to the query, K is the key, V is the value and $softmax$ is a normalization function. Given the input embedding, which symbolize the sentence, its goal is to understand the importance of a word based on its context, i.e., how much attention to

place in each part of the input as we encode a word for a specific sentence position. Q, K and V are matrices that the model learns during the training process, aiming to represent multiple sub-spaces for the same text, i.e., the ability to focus on different positions of the sentence input at the same time.

2.3.2 Neural IR Models for Text Matching

Neural models have been used for several downstream applications in the context of text match, including Question Answering (WAN et al., 2016), Paraphrase Identification (PANG et al., 2016), Ad-hoc Search (XIONG et al., 2017) and Document Retrieval (GUO et al., 2016). Built on the top of convolutional layers, siamese approaches or recurrent network architectures, they capture the correlation between query and document terms based on an interaction step (i.e., a matching matrix extracts relevant similarities from both sides of the input). As follows, we describe the models used in this thesis work. We chose such neural IR approaches since they have been proposed for text-matching problems (GUO et al., 2019), similar to our table/dataset retrieval tasks, which also consider text matching from the metadata information.

- **DSSM** (HUANG et al., 2013). This model maps query and document terms to a low-dimensional space by using a Deep Neural Network (DNN). Its input layers (i.e., the raw text) are represented by a high-dimensional word vector as, e.g., the TF-IDF or the N-Grams for the text inputs, and a set of non-linear layers embeds query and document terms to a dense semantic space, which can be used for semantic similarity in a retrieval task. The relevance score is then calculated by applying the cosine distance over query/document vectors in that semantic space.

- **ARCI** (HU et al., 2015). This model is a siamese-based network that uses 1D convolutional layers to learn semantic representations from text sentences. It takes the word embedding of each part of the input and summarizes it by employing layers of convolution and pooling, in which each vector representation is learned by using an individual branch of the network. The fixed vector for each sentence is then utilized to compute their matching degree over a Multi-Layer Perceptron (MLP) network.

- **ARCII** (HU et al., 2015). This model is an evolution of ARCI and utilizes 1D convolutions to construct an interaction step for the sentences in the input (i.e., a matching matrix), which represents their possible combinations in a low level representation space. However, it further includes 2D convolution layers, over the previous matrix, to encode high-level text similarities. At the top of the architecture, pooling layers and MLP networks are used to compute the matching degree for the text input pair.

- **MVLSTM** (WAN et al., 2016). This model utilizes a Bidirectional Long Short-Term Memory (Bi-LSTM) to learn long and short-term dependencies from both sides

of the input sentences, which are represented by word embedding approaches. In sequence, relevant text matches are captured by using an interaction step over the Bi-LSTM vectors, in which their similarity functions include cosine distance, bilinear operations or a tensor layer. At the top of MVLSTM architecture, pooling layers and a multilayer perceptron network aggregate the interaction matrices to produce the final matching score for the input.

- **DRMM** (GUO et al., 2016). This model is a deep-based network that captures local interaction from the query/document terms based on their neural embeddings, which are then mapped to a fixed-length matching histogram. Similar to the previous approaches, this model also utilizes cosine distance for measuring the semantic similarity from the input. In sequence, the matching histogram, which represents the text similarities according to a discretized set of bins for the local interactions, feeds a forward matching network in order to learn hierarchical matching for query and document. At the top of the architecture, a term gating network aggregates the overall matching to produce the final similarity scores for a query-document pair.

- **MATCH PYRAMID** (PANG et al., 2016). This model is an image recognition-based approach for modeling text matching. Its core idea is to use a matching matrix, which represents the text similarity for words, phrases or sentences, to obtain the semantic correlation from text inputs. This network is similar to the neural models for image detection, which capture patterns from visual elements using distinct image representations as, e.g., 1D and 2D pixel grids. Based on that, the matching degree for text words is calculated by utilizing cosine distance or the dot product function over their word embedding vectors. In addition, a set of convolutional layers is also utilized for producing hierarchical matching patterns from the input. At the top of the network, a set of pooling layers and an MLP architecture produce the final matching score for the text input.

- **KNRM** (XIONG et al., 2017). This model utilizes a kernel-based network for extracting matching features from query and document terms. For that, given the word embedding representations for the input, a translation layer calculates the text correlation based on the cosine distance across their vector embeddings, also producing a translation matrix that defines the word similarities. In sequence, a kernel-polling approach is also utilized to convert word-word interaction to query-document ranking features, whose maps the matching degree for query and document. At the top of this network-architecture, a ranking layer combines the ranking features, obtained from the kernel-polling layers, for producing the final ranking score.

- **CONV-KNRM** (DAI et al., 2018). Similar to KNRM, this model also applies kernel-pooling layers to match queries and documents in a ranking task. However, a dif-

Table 1 – An overview of the neural IR models for text matching that we have explored in this thesis work. We summarize their core architecture in terms of input/output, hidden layers, training strategy and cross-match methodologies.

Model	Input Layer	Hidden Layers	Last Layer	Training Strategy	Cross Match
DSSM Huang et al. (2013)	TF-IDF Word Hashing	Non-Linear	Softmax	Triplet Loss	No
ARC I Hu et al. (2015)	Word Embedding	1D Convolution Max Pooling	MLP	Triplet Loss	Yes
ARC II Hu et al. (2015)	Word Embedding	2D Convolution Max Pooling	MLP	Triplet Loss	Yes
MVLSTM Wan et al. (2016)	Word Embedding	Bi-LSTM Max Pooling	MLP	Triplet Loss	Yes
DRMM Guo et al. (2016)	Histogram Mapping Word Embedding IDF	Feed-Forward Term Gating	Feed-Forward Network	Triplet Loss	Yes
MATCH-PYRAMID Pang et al. (2016)	Word Embedding	2D Convolution Max Pooling	MLP	Binary Classification	Yes
KNRM Xiong et al. (2017)	Word Embedding	Translation Layer Kernel Pooling	Tangent Rank	Triplet Loss	Yes
CONV-KNRM Dai et al. (2018)	Word Embedding	1D Convolution Cross Matching Kernel Pooling	Tangent Rank	Triplet Loss	Yes
DUET Mitra, Diaz and Craswell (2017)	One-Hot-Encoding Character N-Graph	Fully Connected 2D Convolution Max Pooling	Hyperbolic Tangent	Posterior Probability	Yes

Source: Created by the author

ference is that the word embeddings from the text inputs are first mapped to distributed representations of n-gram embeddings by using convolutional layers (i.e., unigrams and bigrams of the text). In addition, a cross-match architecture is also applied to match query/document n-grams, which represents the similarity degree for them. At the top of the network, a kernel-pooling approach utilizes Gaussian kernels to count the soft matches for n-gram pairs, which are then combined into a ranking layer to produce the final matching score for the document.

- **DUET** (MITRA; DIAZ; CRASWELL, 2017). This model uses local and distributed representations of the inputs for matching queries and documents, in which its architecture is composed of two distinct sub-networks, whose parameters are jointly optimized during the training phase. Based on that, the local network estimates the matching degree for query and document terms based on their exact matches. In contrast, the distributed network first learns dense representation for query and document terms by employing layers of convolution and pooling, which are then aggregated into a matching matrix according to an element-wise or Hadamard product. At the top of this dual model, both local and distributed similarities feed a fully connected layer to produce a single score.

We conclude this section by summarizing the models for text match or text retrieval that we previously discussed in Table 1, which we categorize according to input/output

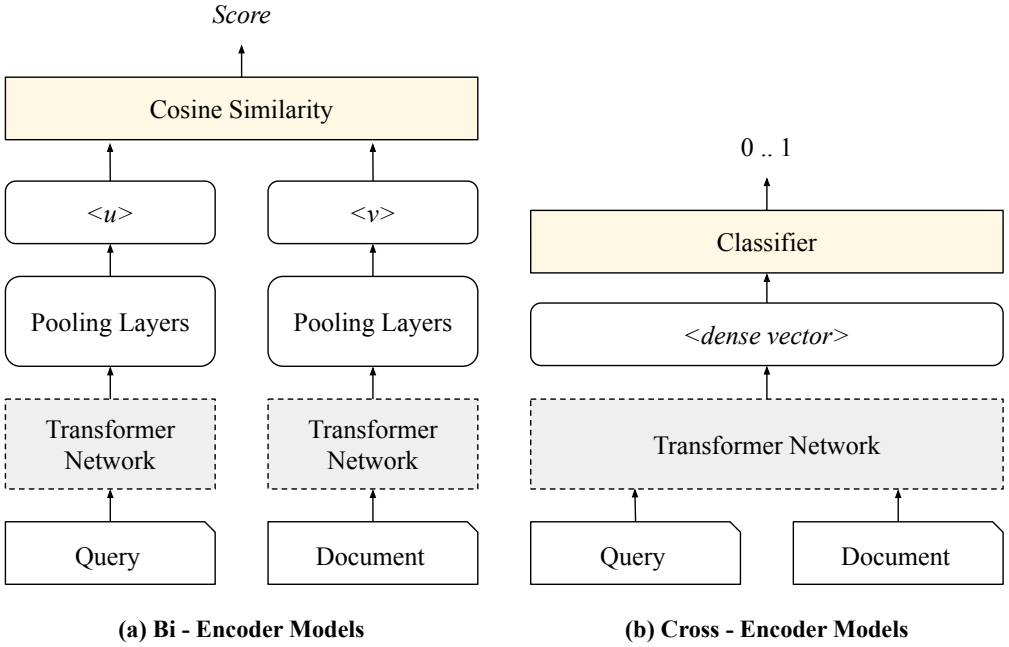
approaches, hidden layers, training strategy and cross-match methodologies. In relation to their input method, several approaches for this task have used word embedding techniques for text representation, which construct dense vectors for query/document terms based on neural networks (ARCI, ARCII, MVLSTM, DRMM, M-PYRAMID, KNRM, CONV-KNRM). Regarding their hidden layers, convolution and max-pooling architectures are most common for them, whose goal is to capture relevant matching signals from both sides of the input (ARCI, ARCII, M-PYRAMID, DUET). Besides, note that a cross-match strategy is also applied to various models, in which an interaction step learns the similarity degree between query and document terms (except for DSSM model). At this front, cosine distance or dot-product is widely utilized for this matching matrix. Lastly, at the top of the networks, MLP architectures have been generally applied to produce the final matching score (ARCI, ARCII, MVLSTM, M-PYRAMID) and, concerning their training strategy, matching models for text retrieval have usually explored a triplet loss method, in which the objective is to minimize the distance from the positive example and its anchor (i.e., a reference input), in addition to maximizing the distance from the negative ones (DSSM, ARCI, ARCII, MVLSTM, DRMM, KNRM, CONV-KNRM).

2.3.3 Neural IR Models for Dense Retrieval

With the popularity of the transformer network, many studies have used this network-architecture as a dense retrieval approach, in which dense vectors, built on the top of the transformer block, encode both query and document terms for ranking. In general, dense retrievers are applied according to two distinct methodologies, which we illustrate in Figure 11: (a) *bi-encoder models*; and (b) *cross-encoder models*. Bi-encoder models embed query and document terms by utilizing distinct transformer networks, whose similarity degree is usually computed over their dense vectors at the top of the architecture, and the cosine distance is widely utilized for this task. An example of a bi-encoder is SBERT, which uses siamese and triplet networks to derive semantic embeddings (REIMERS; GUREVYCH, 2019). In contrast, cross-encoder approaches, perform full attention to the query-document input pair by employing a single transformer network, in which a dense representation feeds a classification layer, on the top of architecture, to generate the final matching label. An instance of a cross-encoder is BERT, a sentence encoder that uses bidirectional transformer networks for language representations (DEVLIN et al., 2019).

At this direction, in this thesis work, we have evaluated the models: Dense Passage Retrieval (DPR) (KARPUKHIN et al., 2020) and Sentence-BERT (SBERT) (REIMERS; GUREVYCH, 2019). DPR is a two-independent neural network for open-domain Question Answering. It first maps text passages to dense vectors by employing a passage encoder based on the BERT architecture, also building an index for retrieval. In addition, at inference time, a query encoder is also utilized for embedding the input query, similar to the passage encoder. As a result, queries and text passages are then aggregated by using the

Figure 11 – Neural-based networks for dense retrievers in two architectures: (a) bi-encoder models, in which queries and documents are individually mapped to dense vectors by using distinct transformer networks; and (b) cross-encoder models, whose a single transformer network jointly encodes query and document terms to a unique vector representation, which feeds a classification layer.



Source: Adapted from SBERT (REIMERS; GUREVYCH, 2019)

dot-product similarity as a ranking function for passage retrieval. In our setup, we use a pre-trained DRP model that was fine-tuned in Google's Natural Questions dataset.[1]

Similar to DRP, SBERT is another BERT-based model for generating fixed-sized vectors from input text sentences, which can also be compared by using cosine distance in a retrieval task. One difference is that SBERT uses two distinct network architectures to derive semantically embeddings: (a) siamese networks, which compute the element-wise difference for query and document terms employing cosine similarity; and (b) triplet networks, whose goal is to approximate an anchor sentence to a positive sentence by employing a triplet loss function. SBERT was fine-tuned on 1B training pairs using multiple datasets for semantic search tasks.

2.4 EVALUATION MEASURES IN INFORMATION RETRIEVAL

Several evaluation measures have been used in IR to assess the performance of the text retrieval approaches including *Precision*, *Recall* and *F-Measure*, which usually compute the fraction of relevant documents retrieved for a query, in addition to evaluating the search accuracy (MITRA; CRASWELL, 2018). Besides, there also exist more specific ranking

[1] https://ai.google.com/research/NaturalQuestions

metrics such as: Mean Reciprocal Rank (MRR) (CRASWELL, 2018), which measures the inverse rank position for the first correct answer over a sample of queries; Mean Average Precision (MAP) (ZHU, 2004), whose represents the average precision for a ranked list of documents; and Normalized Discounted Cumulative Gain (NDCG) (JÄRVELIN; KEKÄLÄINEN, 2002), which assesses the effectiveness of a ranking algorithm based on relevance judgment for a query-document pair.

In the context of this thesis work, we measure *Recall* to evaluate the quality of our retrieval approach (i.e., the Elasticsearch retriever for the *top*@100 candidate tables/-datasets), which we calculate according to Equation 2.5:

$$Recall = \frac{TP}{TP + FN} \tag{2.5}$$

where True Positive (TP) represents the matching tables/datasets that were recovered by the retriever and False Negative (FN) is the ones that are not in the retrieval set. Moreover, we evaluate our *News-Table* matching model by using MRR, according to Equation 2.6:

$$MRR = \frac{1}{|Q|} * \sum_{i=1}^{|Q|} \frac{1}{rank_i} \tag{2.6}$$

where $|Q|$ represents the total of queries for the test set and $rank_i$ refers to the rank position of the first relevant document for the $i - th$ query. For the news-table matching task, we also use accuracy@k (a.k.a. top-k accuracy), which measures the percentage of news articles in the test set correctly matched to at least one of the top-k ranked tables. Accuracy@k is a metric widely used in the evaluation of Question Answering tasks (SANTOS et al., 2015) and recommendation systems (MAITY et al., 2019), for which, like for our problem, there are one or few relevant results for each query. Lastly, regarding NDCG, which measures the quality of the ranking based on relevance judgments, we compute it according to Equation 2.7:

$$NDCG_p = \frac{DCG_p}{IDCG_p} \tag{2.7}$$

where Discounted Cumulative Gain (DCG) is a penalized score for a relevant document based on its ranking position, Ideal Discounted Cumulative Gain (IDCG) defines the perfect document ranking for a query and p means the ranking cut-off for evaluation. The core idea of NDCG is that relevant documents should occur in the first-ranking positions.

2.5 LLMS FOR TEXT DATA AUGMENTATION

With the popularity of the Large Language Models (LLMs), many studies have utilized them to a couple of downstream problems. From zero-shot approaches to few-shot learners (SANH et al., 2022; OUYANG et al., 2022), LLMs have been achieving the best performance for a set of tasks including Machine Translation (BROWN et al., 2020), Passage

Retrieval (GAO et al., 2022), Question-Answering (SANTOS et al., 2020), or Text Summarization (ZHANG et al., 2023). In addition, a novel research area has emerged on applying LLMs for text data augmentation, since there exists a limited number of labeled data in several NLP tasks. In the context of this thesis work, we also focus on LLMs for text query generation, similar to previous studies (BONIFACIO et al., 2022; DAI et al., 2022; JERONYMO et al., 2023; SAAD-FALCON et al., 2023; SACHAN et al., 2022). For example, Bonifacio et al. (2022) propose Inquisitive Parrots for Search (InPars), which is a few-shot method that creates relevant matching queries for text documents by using GPT-3, later used to fine-tune retrieval models in a target task. Another work is presented by Nogueira et al. (2019), in which a Doc2Query approach is applied to generate synthetic queries for text documents, and its goal is to expand the document corpus by incorporating a text query. The core of the Doc2Query strategy is to utilize a transformer-based sequence-to-sequence models for constructing a query given a text document.

Besides, Dai et al. (2022) introduce *PROMPTAGATOR*: Prompt-base Query Generation for Retriever, which also leverages LLMs as a few-shot query generator, in addition to creating task-specific retrievers based on the generated data for ranking. The key idea of PROMPTAGATOR is to produce training samples by prompting a LLM, whose LLM data is then used for fine-tuning text retrieval approaches. At the same front, Sachan et al. (2022) also apply a LLM, T5 or GPT-neo, for question generation in a zero-shot way, in which documents are ranked by computing the likelihood of artificial questions conditioned on the target passages, allowing for dataset-independent re-ranking. In summary, there is an increasing tendency to apply LLMs for text data augmentation. Towards this front, in this thesis work we use them for the dataset retrieval task. As follows, we provide some background about LLMs (Section 2.5.1), in addition to detailing *DocTTTTTquery*, which has been used in this study for query generation (Section 2.5.2).

2.5.1 Large Language Models

Since the announcement of the paper *Attention is All You Need* by Google, transformer networks have gained huge popularity (VASWANI et al., 2017). Inspired by this architecture, LLMs are transformer-based models for language understanding, which are usually composed of an encoder and a decoder block, trained on a massive corpus of data. LLMs are also considered a particular category of sequence-to-sequence models for text generation/representation, and one advantage is that they can also be utilized in an unsupervised manner, in which prompting strategies teach the model output. A prompting strategy is an input instruction for the LLM prediction. For example, we can ask the LLM to summarize a text document or to translate from one language to another.

At this direction, LLMs are mostly applied according to the three following unsupervised methodologies, which we illustrate in Figure 12 (BROWN et al., 2020): (a) zero-shot, in which the LLM predicts the output given a description of the target task; (b) one-shot,

Figure 12 – Prompting strategies for Large Language Models by using three different methodologies: (a) zero-shot; (b) one-shot; and (c) few-shot.

Source: Adapted from Brown et al. (2020)

whose input for the LLM is an example of the task, in addition to its description; and (c) few-shot, which feeds the LLM by employing few samples of the problem and the task description. In summary, zero, one or few-short methods are prompting strategies for the LLM prompting output, whose goal is to guide the text generation in a target task.

2.5.2 DocTTTTTquery Model

In this thesis work, we have used DocTTTTTquery for synthetic query generation, which is an extension of the Doc2Query model (NOGUEIRA et al., 2019).[2] It is a T5-based network architecture for producing artificial questions from text documents. Given an input document, it creates synthetic queries for which the document might respond or even be relevant. DocTTTTTquery was fine-tuned on the MSMARCO Passage Dataset, which contains over 500k search queries for passage documents collected from Bing,[3] and its original motivation was to expand the text document corpus by appending the LLM predicted queries to them, which are indexed for passage retrieval in the target task.

The name DocTTTTTquery stands for a T5-based model. T5 is an encoder-decoder model for language modeling (RAFFEL et al., 2020). Built on top of the Transformer network, its core idea is to convert each task domain to a text-to-text format, where an input text feeds the model, which is then asked to produce some text output. In addition, a

[2] https://github.com/castorini/docTTTTTquery
[3] https://www.bing.com

task-specific (text) prefix is also included in the input sentence, which specifies the target task for the text prediction. For instance, an example of a task prefix is: *translate English to German: That is good.* T5 is inspired by a transfer learning methodology and has been pre-trained on the Colossal Clean Crawled Corpus,[4] which is an English text collection crawled from HTML web pages (about 750 GB of data). This model can also be used for a set of downstream tasks including Machine Translation, Question Answering, Text Summarization, and Text Classification.

2.6 CONCLUDING REMARKS

In this chapter, we presented the IR background in the context of this thesis work: *text retrieval* and *text match*. Based on that, we began by defining IR tasks for ad-hoc document retrieval and question-answering, also explaining IR methods for text retrieval as TF-IDF and BM25. In sequence, we focused in Neural Information Retrieval, in which we introduced the neural IR models for text representation (*Word2Vec*, *Doc2Vect*, *USE*, *BERT*). Moreover, we also defined a set of neural matching models for QA (DSSM, ARCI, ARCII, MVLSMT, DRMM, MATCH-PYRAMID, KNRM, CONV-KNRM and DUET), and dense retrievers, DPR and SBERT, were also demonstrated. Besides, we described the IR evaluation metrics that we apply in our experiments. Lastly, we concentrated on text data augmentation, whose goal was to use LLMs for query generation.

At this direction, the core of our study is to use this background to cover both points of query/document representation and query/document match in our table/dataset retrieval tasks. Furthermore, we also focus on neural models for learning matching patterns from the query-table pair, beyond considering word embedding techniques for their representation. Lastly, our matching solutions are build on the top of the relevant finds from the text matching models we presented in this chapter.

[4] https://www.tensorflow.org/datasets/catalog/c4

Part I

Table Retrieval

3 A SURVEY ON INTELLIGENT SOLUTIONS FOR TABLE RETRIEVAL

Web tables comprise a vast corpus of online relational information. In addition to representing complex data, they also allow a quick understanding of entity relationships. As a result of that, a growing body of work has begun to explore web tables for several downstream applications. For instance, web tables are widely used for the task of Question-Answering (QA), whose goal is to retrieve a table that answers a query from a table collection. Towards this front, a plethora of strategies for table retrieval have been proposed including basic IR methods, entity-based models and neural networks architectures. However, to the best of our knowledge, no previous study has summarized the state-of-the-art or outlined representative solutions for this task. An early survey covers challenges regarding web table extraction, interpretation and augmentation, but presents a limited number of approaches for table retrieval as, e.g., novel deep learning models are not contemplated in the paper (ZHANG; BALOG, 2020). In this chapter, we cover this gap by introducing a survey on table retrieval techniques for QA based on 16 core studies. Specifically, our main contribution is a new table retrieval taxonomy that classifies the table retrieval solutions into five groups, from probabilistic approaches to novel deep learning models. Furthermore, since the tasks of news-table matching and dataset retrieval have a restricted number of studies, we also consider this survey as our related work for this thesis, since both domains use relational tables to represent information. As follows, we start this chapter by providing some background about table retrieval for Question-Answering and Web Tables.

3.1 BACKGROUND

This section covers the table retrieval background. We first formalize the table retrieval problem. In sequence, we present the anatomy of tables on the Web.

Problem Statement. In this chapter, we formalize the QA table retrieval task as follows: given a query q_i and a set of web tables $T = \{t_1, t_2, t_3, ..., t_n\}$, our goal is to find the most relevant table t_i to q_i. The notion of relevance is broad, since a table can summarize the query, bring contextual data or even answer upcoming questions. Formally, we assume the task of table retrieval for QA as a ranking task, i. e., the objective is to learn a scoring function $f : q_i \times t_j \mapsto \mathbb{R}$ that scores tables in T for a given text query q_i to rank them based on the table aspects.

Table Anatomy. Web tables are contained in web pages and defined by a set of features as follows (ZHANG; BALOG, 2020): (a) *page title* (HTML title), (b) *surrounding text* (page's short description), (c) *table headers*, (d) *table body* and (e) *table caption*. The first two ones describe information around the table (i.e., reference text) and have been widely

Figure 13 – An example of a web table with its five aspects and a core-column: (a) page title, (b) surrounding text, (c) headers, (d) body and (e) caption. The 2nd column shows the core-column.

List of Most Expensive Paintings (a) Page Title

This is a list of the highest known prices paid for paintings. The current record price is approximately US$450.3 million paid for da Vinci's *Salvator Mundi*. **(b) Surround Text**

Price	Painting (*Core Column*)	Artist	Year	Date of Sale
$450M	Salvator Mundi	Leonardo da Vinci	1500	11/2017
$300M	Interchange	Willem de Kooning	1955	09/2015
$250M	The Card Players	Paul Cézanne	1892	04/2011

(c) Headers **(d) Body**

***List of Highest Prices Paid* - (e) Caption**

Source: Adapted from the original web page

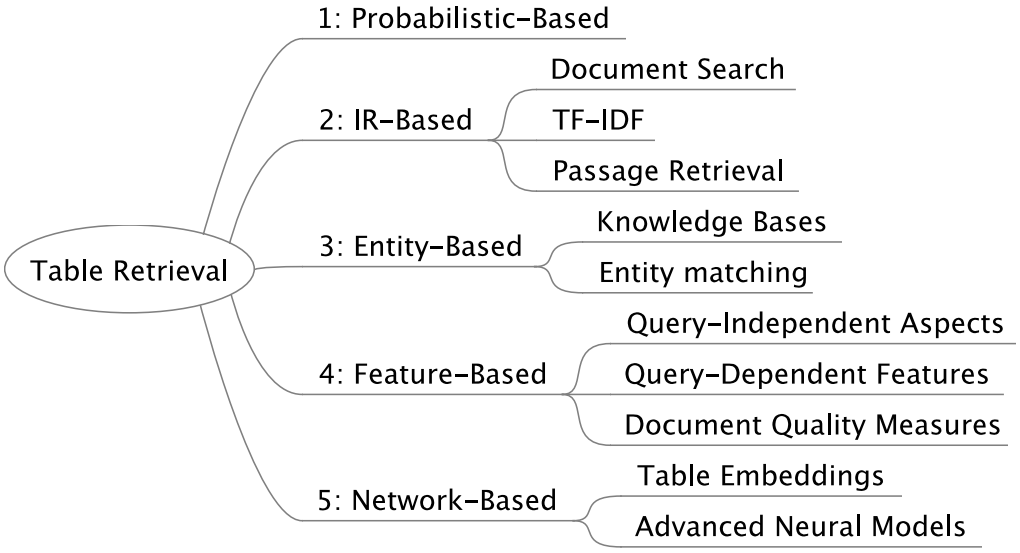
used in table retrieval approaches for this task (CHEN et al., 2020b; LIU et al., 2007b; PIMPLIKAR; SARAWAGI, 2012; VENETIS et al., 2011; ZHANG; BALOG, 2018; ZHANG; ZHANG; BALOG, 2019). The header indicates the properties of the column and helps to describe their meaning. The body (i.e., cells) contains all the table content, and the caption describes the subject of the table data. In addition, some tables often include a core column, which depicts real-world entities or particular categories, while the remaining ones represent specific proprieties about them. Lastly, these table's cells contain natural language sentences or numerical values. We demonstrate a concrete example of the table anatomy in Figure 13: a Wikipedia table that lists the most expensive paintings in the world.[1]

3.2 A TABLE RETRIEVAL TAXONOMY

This section presents our table retrieval taxonomy for the task of question-answering. Note that our survey includes studies for this task published up to the year 2020. Moreover, we adopt a snowballing methodology for the literature review, in which our start set is the work presented by Zhang and Balog (2018). We select those studies that deeply describe their table retrieval methodology. At this front, the proposed classification is shown in Figure 14. In order to summarize the best solutions for this task as well as to investigate representative methods for calculating the correspondence between unstructured queries and structured tables, we categorize the state-of-the-art in at least five fronts according to the type of methodology: (1) *Probabilistic-Based* (GAO; CALLAN, 2017; PYREDDY; CROFT, 1997; SHRAGA et al., 2020b), which utilize prior and posterior probabilities to estimate the query/table similarity; (2) *IR-Based* (CAFARELLA et al., 2008b; LIU et al., 2007a; LIU et al.,

[1] https://en.wikipedia.org/wiki/List_of_most_expensive_paintings

Figure 14 – Our taxonomy for QA table retrieval. We categorize the proposed solutions in five fronts: (1) probabilistic-based, (2) IR-based, (3) entity-based, (4) feature-based and (5) networks-based.



Source: Created by the author

2007b; PIMPLIKAR; SARAWAGI, 2012; SHRAGA et al., 2020a), whose goal is to apply basic methods for document search to retrieve the tables; (3) *Entity-Based* (SUN et al., 2016; VENETIS et al., 2011; ZHANG; BALOG, 2018), which find co-related tables based on entity matches over the table data; (4) *Feature-Based* (BHAGAVATULA; NORASET; DOWNEY, 2013; CAFARELLA et al., 2008b; CHAKRABARTI et al., 2020), where the objective is to use query-dependent and query-independent features for training regression models; and (5) *Network-Based* (CHEN et al., 2020b; GLASS et al., 2021; SHRAGA et al., 2020c; SUN et al., 2019; TRABELSI; DAVISON; HEFLIN, 2019; ZHANG; ZHANG; BALOG, 2019), which learn query-table context vectors by adopting neural networks architectures. We should note that each article belongs to only one front, except for Cafarella et al., (2008), which cover both IR methods and feature-based approaches in the same article. As follows, we overview these solutions according to the devised taxonomy.

Probabilistic-Based. To the best of our knowledge, the earliest studies of table retrieval target this problem by utilizing probabilistic models (GAO; CALLAN, 2017; PYREDDY; CROFT, 1997), and popular applications for this task include INQUERY System (CALLAN; CROFT; HARDING, 1992) and Indri Seach Engine (METZLER; CROFT, 2004) (common frameworks for document search). In this front, single queries are represented in a structured way by identifying the most important query concepts such as entities or noun phrases. In addition, tables are indexed as documents by using distinct index fields (e.g., table caption, headers and body), and each of them can also receive different weights in the retrieval score based on user knowledge (PYREDDY; CROFT, 1997). Lastly, these solutions

estimate the similarity between queries and tables by employing adapted mathematical functions composed of prior and posterior probabilities (GAO; CALLAN, 2017).

IR-Based. Some studies propose to query existing web search engines to retrieve tables from the best-ranked documents (CAFARELLA et al., 2008b). In another direction, some approaches search for tables directly into a table corpus (LIU et al., 2007a; LIU et al., 2007b; PIMPLIKAR; SARAWAGI, 2012; SHRAGA et al., 2020a). For this front, tables are generally interpreted as single or multi-field (text) documents. Beyond that, in a table multi-field approach, these studies also use the table-structure to consider different meanings. For example, the table caption is a more descriptive aspect as it outlines the whole table data and therefore can receive more weight at the score function. Overall, such class of work focuses on the lexical matching between queries and tables (i.e., query/table terms overlap), and basic IR methods for document search such as TF-IDF are widely used for this task (LIU et al., 2007a). Finally, tables are ranked by using cosine similarity over TF-IDF or by applying BM25 algorithm (SHRAGA et al., 2020a).

Entity-Based. Beyond lexical matching, other studies have used knowledge bases to extract semantic features for this task (GAO; CALLAN, 2017; ZHANG; BALOG, 2018), aiming to compute the semantic correlation between the queries and the table data. Towards this direction, a semantic solution for table retrieval is to find those tables that match entity-related terms on the query over a table core column, which represents the main column of the table (CHAKRABARTI et al., 2020; SUN et al., 2016; VENETIS et al., 2011). Besides, other papers propose robust models in which the query and the table are represented by distinct semantic spaces (e.g., bag-of-concepts or embeddings), and the similarity between them is calculated according to novel early and late fusion strategies (ZHANG; BALOG, 2018).

Feature-Based. When human-curated data is available, several studies have been applying linear regression models to produce a similarity score between queries and tables (BHAGAVATULA; NORASET; DOWNEY, 2013; CAFARELLA et al., 2008b; CHAKRABARTI et al., 2020). To train such approaches, the literature presents a set of query-dependent and query-independent features, which mostly include query/table terms overlap and document measures. For instance, there exist papers that consider the number of times that a query-term occurs in the most-left table columns (CAFARELLA et al., 2008b; CHAKRABARTI et al., 2020). Moreover, other authors have argued that the table retrieval results improve when using external document aspects into these models, such as page rank, page views or in/out reference links for the page, i.e., document quality measures (BHAGAVATULA; NORASET; DOWNEY, 2013).

Network-Based. As neural networks have been applied for a range of tasks, some articles also adopt those strategies for table retrieval (CHEN et al., 2020b; GLASS et al., 2021; SHRAGA et al., 2020c; SUN et al., 2019; TRABELSI; DAVISON; HEFLIN, 2019; ZHANG; ZHANG; BALOG, 2019). For instance, previous studies apply word-embedding techniques to embed query-table terms (TRABELSI; DAVISON; HEFLIN, 2019; ZHANG; ZHANG; BALOG, 2019).

In another direction, recurrent networks have also been used to generate query/table context vectors (SUN et al., 2019). In addition to that, other papers also represent queries and tables as multimodal objects by employing sophisticated deep neural architectures based on gated units (SHRAGA et al., 2020c), and recent studies have also utilized novel bidirectional sentence encoders for this task such as BERT and ALBERT (CHEN et al., 2020b; GLASS et al., 2021).

On the whole, this flurry of research points out several alternatives for the problem of table retrieval in the context of QA, ranging from probabilistic approaches to novel sophisticated deep neural-network architectures.

3.3 REPRESENTATIVE METHODS FOR TABLE RETRIEVAL

In this section, we detail representative methods for the task of table retrieval in which fall under one of the categories of our taxonomy.

3.3.1 Probabilistic Approaches

The early study of table retrieval we found is presented by (PYREDDY; CROFT, 1997), where the authors introduce *TINTIN*: a retrieval system for tables in text documents. *TINTIN* has three main components: table extraction, annotation and retrieval. Specifically for table retrieval, the authors use the *INQUERY* System (CALLAN; CROFT; HARDING, 1992), and tables are indexed based on their caption, headers and cells. *INQUERY* is a probabilistic retrieval engine based on Bayesian Inference Networks: a Directed Acyclic Graph (DAG) whose nodes are propositional variables and arcs are dependencies. For this task, both tables and queries are nodes, and arcs represent the probability of a table responding to a query.

At the same direction, Gao and Callan (2017) propose *MaitreD*: a probabilistic ranking framework for scientific table retrieval. Unlike *TINTIN*, which only uses basic table aspects as indices (i.e., caption, headers and cells), *MaitreD* brings out novel features for this task including the table-quality, the query-concepts and the quantity-terms. The first one evaluates the quality of the table-data by counting the total of numerical cells inside the table body. Another aspect is the query representation, *MaitreD* transforms unstructured queries into structured queries by analyzing the type of query concepts (e.g., entities, noun phrases and quantity expansion). Entities are annotated by TagMe (CARMEL et al., 2014), a popular system for annotating short texts with Wikipedia entities, and noun phrases are identified by applying Monty Lingua.[2] In addition, quantity features map query terms to quantity descriptions and standard units (e.g., electron volt and joule) by employing *QUDT*: a data type ontology.[3]

[2] http://alumni.media.mit.edu/ hugo/montylingua
[3] http://www.qudt.org

3.3.2 Traditional IR Methods

This section details the IR methods for table retrieval. We classify them into Document Search, TF-IDF and Passage Retrieval approaches.

Document Search. Cafarella et al. (2008b) introduce two basic methods for QA table retrieval: *NaiveRank* and *FilterRank*. The first one searches for the queries on the top of an existing web search engine, and tables are extracted from the top-k ranked documents. If there is more than one table per web page, it ranks the tables in the order where they appear in the document (top-down). *FilterRank* is slightly different compared to *NaiveRank* as it scans down the search engine's results until obtains k tables.

TF-IDF. Another study in this front is presented by Liu et al. (2007a), Liu et al. (2007b), where the authors introduce *TableRank*: an algorithm adapted from TF-IDF which weighs terms by using three levels: Table Term Frequency - Inverse Table Term Frequency (TTF-ITTF), Table Level Boosting (TLB) and Document Level Boosting (DLB). TTF-ITTF measures term frequency over a table metadata instead of the whole corpus for preventing false-positive hits. TLB goes over two table-level features: *table-frequency*, which denotes the total number of tables that contains a term in the entire document; and *table document position*, which weights table terms when they appear over sections specified by the user (e.g., "Experiment", "Evaluation" and "Result Analysis"). Concerning DLB, it addresses query-independent features such as the overall importance of a document where a table occurs. The authors claim that the tables present in important documents are also inclined to be relevant. Lastly, the final vector is computed by aggregating the query/table terms in TTF-ITTF, TLB and DLB levels, and the matching score between queries and tables is calculated by the cosine similarity.

Passage Retrieval. Whereas Cafarella et al. (2008) and Liu et al. (2007) use traditional methods for document search to retrieve tables, Shraga et al. (2020a) introduce two novel measures for this task: *Intrinsic* and *Extrinsic* query/table similarities. In this work the authors assume there is a pool of candidate tables obtained by some traditional IR model, which they re-rank according to the proposed similarities. The *Intrinsic* score for a query-table pair is based on techniques for passage-retrieval (ROITMAN; MASS, 2019) and is computed according to the following steps: (i) tables are represented as the concatenation of all textual aspects (i.e., page title, table caption, headers and body); (ii) the whole text is divided into k passages by using a sliding window over characters; (iii) a score is calculated for each of them by combining query-dependent and query-independent measures built on BM25 and a modified TF-IDF cosine similarity; and (iv) the maximum score from all passages is the intrinsic similarity of a table. Another proposal is the *Extrinsic* score, which assumes the cluster hypothesis in Information Retrieval, i.e., similar documents tend to behave similarly for a given information need (KURLAND, 2013). Based on that, the authors argue that the table-table correlation goes analogous

and compute the inter-table similarity for a pair of tables by adopting the Bhattacharyya coefficient (BHATTACHARYYA, 1946). Lastly, the authors re-ranking the set of candidate tables by multiplying the scores from the IR model, intrinsic and extrinsic similarities to create the final score of each table.

3.3.3 Entity-Based Models

This section presents the entity-based models for table retrieval, which we classify into Knowledge Bases and Entity Matching approaches.

Knowledge Bases. Zhang and Balog (2018) introduce a semantic similarity model in which the query and the table aspects are represented by distinct semantic spaces (i.e., bag-of-concepts and word embeddings). Specifically, the proposed model contains three cascade steps: (i) content extraction, (ii) semantic modeling and (iii) table matching. The first one represents the query/table terms as words or entities. The entities are extracted from DBpedia[4] knowledge base, whose features include name, attributes and similar/related entity names. The second step goes over two kinds of semantic representations: bag-of-concepts (Wikipedia entities or categories); and embeddings, which map each table/query term to a dense vector using Word2Vec (MIKOLOV et al., 2013b) and RDF2Vec (RISTOSKI; PAULHEIM, 2016). The final step of the model measures the similarity between queries and tables according to two strategies: early fusion, which encodes query and table as a single representation based on the centroid vector; and late fusion, which computes the pairwise similarity between all query and table terms, and an aggregation function consolidates the similarity (e.g, *sum*, *max* or *avg*).

Entity Matching. In the same direction, other studies have also been using entity-based approaches for matching queries and structured Web Tables (SUN et al., 2016; VENETIS et al., 2011). However, such solutions are quite simpler compared to Zhang and Balog (2018) because they only focus on entity matches over the table features (generally table cells). For example, Venetis et al. (2011) target this problem by annotating table-columns as entities or categories. The queries are represented as a tuple of $\langle$*class name*, *property*$\rangle$, which denotes query entities and their relationships. To retrieve the co-related tables for a user query, their retrieval solution finds the tables that match the query *class name* over the table data. Another work is presented by Sun et al. (2016), which also fetches the tables according to query entity matches inside the table columns.

3.3.4 Feature-Based Techniques

Cafarella et al. (2008b) introduce two table retrieval approaches in which they train a linear regression estimator to produce a query-table score by applying query-independent and query-dependent features: *FeatureRank* and *SchemaRank*. The first one uses the fol-

[4] https://www.dbpedia.org/

lowing aspects for training the model: (i) *Table Data*, ⟨#rows, #columns, #nullvalues⟩; (ii) *Document Quality Measures*, ⟨page rank⟩; and (iii) *Query/Table Terms Overlap*, #hits on table ⟨headers, columns, cells⟩. *SchemaRank* is similar to *FeatureRank* but also includes the ACSDb-based coherency score as a novel retrieval aspect. A coherent schema is one where the attributes are tightly related to one another, i.e., how well the attributes fit together. For this task, the coherence level among table columns is computed by using Point-wise Mutual Information (PMI) (CHURCH; HANKS, 1989).

In the same direction, Bhagavatula, Noraset and Downey (2013) propose to train regression models to retrieve tables for a query but with some main differences compared to Cafarella et al. (2008). First, the solution extracts tables from the top-30 web pages returned by a traditional engine for a given query. They also propose novel features as input for the models including: the total of in/out links to the page; table importance; and the number of query tokens found in page title or table caption.

Another study towards this front is presented by Chakrabarti et al. (2020), which utilizes similar features as in Bhagavatula, Noraset and Downey (2013) and Cafarella et al. (2008b). In addition, they propose a set of novel statistical aspects based on entity matches and pre/post query modifier terms for this task, and a gradient boosted decision tree is used for training instead of a linear regressor. The authors argue that each table has a subject column, which typically contains entity names, and therefore the proposed features include the total of query entity matches over the *subject column name* or *subject column values*, and the number of matches between query entities and document headings (i.e., h2, h3, and h4). Lastly, this study also claims that for some type of queries there is a *premodifier* or *postmodifier* term. For instance, the query ⟨*2017 tom cruise movies*⟩ has a *premodifier* term, i.e., the token 2017, which limits the search to this year. Based on that, the authors argue that such tokens generally do not occur in the page features or subject columns and therefore they devised another statistical metric for counting the frequency in which this terms appear over *non-subject columns* of the tables.

We conclude this section by categorizing the most common query/table features applied to regression models for this task in these studies: query dependent and query independent aspects; and document quality measures, summarized in Table 2. We depict feature names, short description and article source.

3.3.5 Neural Networks

This section details neural models for table retrieval. We present Table Embeddings approaches and Advanced Neural Models.

Table Embeddings. Zhang, Zhang and Balog (2019) propose *Table2Vec*: a neural language modeling for embedding tabular data into distinct vector spaces. Specifically, they introduce four types of table embedding: *Table2VecW*, *Table2VecH*, *Table2VecE* and *Table2VecE**. *Table2VecW* considers words over the page title, section title and table ele-

Table 2 – Common features for training regression models. We classify them into three groups: Query Independent, Query Dependent and Document Measures.

Feature	Description	Bhagavatula (2013)	Cafarella (2008a)	Chakrabarti (2020)
(1) - Query Independent				
#Rows	Total of Rows	x	x	x
#Columns	Total of Columns	x	x	x
#NullValues	Ratio of Empty Cells	x	x	x
Has-header?	If Headers Present		x	
SectionNumber	Wikipedia Section	x		
TableImportance	Total of Tables by Page	x		x
TablePageFraction	Table Size Ratio	x		x
ColumnName	If Column Names Present			x
(2) - Query Dependent				
#Hits on header	Query Hits on Headers		x	
#Hits on Columns	Query Hits on Columns		x	x
#Hits on Body	Query Hits on Cells		x	
qInPgTitle	Query Hits on PgTitle	x		x
qInTableCaption	Query Hits on Caption	x		x
qInColumnNames	Query Hits on Col-Name			x
qInSurrText	Query Hits on SurrText			x
(3) - Document Measures				
Page Rank	Document Search Rank		x	x
ACSDb Schema	Headings Coherency		x	
YRank	Yahoo Rank	x		
inLinks	Total of In-Links to Page	x		
outLinks	Total of Out-Links From	x		
PageViews	Total of Page Views	x		

Source: Created by the author

ments (i.e., caption, headers and cells), while *Table2VecH* only encodes table header terms. *Table2VecE* looks at all entities that appear inside the table cells, and *Table2VecE** creates embeddings based on entities names over a table core column. Although *Table2Vec* also examines entity-based methods for table retrieval similar to those studies presented in Section 3.3.3, the main goal is to encode tabular data built on top of neural networks. For this task, the embeddings are created by using an adapted skip-gran model based on *Word2Vec* (MIKOLOV et al., 2013b), and the cosine distance is applied to compute the similarity of a query-table pair.

Similarly, Trabelsi, Davison and Heflin (2019) propose a solution that focuses on table embeddings considering three stages. First, it builds an embedding model where token embeddings are calculated by using the contextual information from each table (table metadata such as title or caption; table headers; and table body). By doing this, a token that appears over a table metadata has a different meaning from the same token when it is inside the table. The second stage augments the table tokens by adding their similar top-k words, which are computed by the built embedding model. Lastly, a cosine-based similarity measures the level of matching between query and table tokens.

Advanced Neural Models. Sun et al. (2019) introduce a table retrieval model based on two cascaded steps. First, it finds a small set of candidate tables by using traditional IR

approaches such as BM25 (i.e., 50 or 100 candidate tables). Afterward, the model employs a neural network to automatically extract features from the initial set as well to produce the similarity score. The main contribution is a set of both manually designed characteristics and neural network features for this task. The designed characteristics include: *word-level*, the total of query terms that overlap inside the table aspects (i.e., header, cells and caption); *phrase-level*, which uses a Statistical Machine Translation (SMT) (KOEHN; OCH; MARCU, 2003) to extract a phrase table from a bilingual corpus and deals with those cases where queries and tables use different expressions but have the same meaning; and *sentence-level*, which applies the CLSM model (SHEN et al., 2014) to compute sentence vectors from sub-word embedding by utilizing convolutional neural networks. Beyond that designed characteristics, the authors also propose network-based features created by recurrent networks for encoding queries and tables. A bidirectional Gated Recurrent Unit (GRU) (CHO et al., 2014b) encodes the queries from both directions, and the last two hidden states of the model architecture are used as the final embedding vector. In addition, because tables are composed of many aspects as headers and cells, their approach represents table features as external embedding memories, which associates an embedding for each table token.

Chen et al. (2020b) propose a deep contextualized language model based on BERT (DEVLIN et al., 2019). The retrieval framework contains four components as follows. First, a content selector extracts potentially informative items from each table by slicing table aspects into a list of rows, columns and headers. The second component selects relevant items based on a salience score, which measures the similarity between query and table terms by applying cosine distance. In addition, Chen et al. (2020b) introduce a set of three content selectors to encode table items into a fixed-width BERT representation based on *mean*, *sum* and *max* metrics. Next, the solution uses BERT to encode queries and those table items with higher salience scores (third component). The last component concatenates BERT features with curated aspects into a single vector. The authors argue that curated-based methods have shown impressive performance, and the proposed framework can also include human-curated features when they are available (such aspects are similar to that we show in Section 3.3.4). Lastly, a final regression layer predicts the similarity score for a query-table input pair.

Another work, by Shraga et al. (2020c), introduces a novel deep learning model for table retrieval. The tables are represented as multimodal objects, i.e., each table $T_i = \langle description,\ schema,\ records,\ facets \rangle$. The description is the text that accompanies the table such as page title or caption, while the schema is the table column names. Records consist of the table rows, and facets vertically divide the table data and correspond to a specific table column. The proposed approach encodes each of those table modalities by applying different types of neural networks. Specifically, the solution uses a bidirectional Long Short-Term Memory (LSTM), followed by a max-pooling layer, to represent both

Table 3 – Table retrieval solutions classified by year of publication. We show common methodologies by reference and table aspects for this task. The symbol (*) indicates the techniques or the type of features chosen by each approach.

Reference	Methodology	Single Field	Multi Field	Entity Modeling	Surrounding Features	External Aspects
Pyreddy and Croft (1997)	Probabilistic-Based	*	*			
Gao and Callan (2017)	Probabilistic-Based		*	*	*	
Liu et al. (2007a)	IR-Based	*			*	*
Pimplikar and Sarawagi (2012)	IR-Based	*			*	
Shraga et al. (2020a)	IR-Based		*		*	
Venetis et al. (2011)	Entity-Based		*	*	*	*
Sun et al. (2016)	Entity-Based	*		*		
Zhang and Balog (2018)	Entity-Based	*		*	*	
Cafarella et al. (2008b)	Feature-Based	*	*			*
Bhagavatula, Noraset and Downey (2013)	Feature-Based		*		*	*
Chakrabarti et al. (2020)	Feature-Based		*	*	*	*
Sun et al. (2019)	Neural Networks		*			
Zhang, Zhang and Balog (2019)	Neural Networks	*		*	*	
Trabelsi, Davison and Heflin (2019)	Neural Networks		*		*	
Chen et al. (2020b)	Neural Networks		*		*	
Shraga et al. (2020c)	Neural Networks		*		*	

Source: Created by the author

query and table description. For the table schema, a Multi-Layer Perceptron (MLP) block generates a representation for each table header. Regarding table records and facets, the authors propose a 3D Convolutional Neural Network (CNN), followed by max-pooling layers. Another main contribution is how to obtain a joint representation from all these table modalities. For that, the authors apply a Gated Multimodal Unit (GMU) (OVALLE et al., 2017): a gating scheme for learning a single representation of distinct table modalities.

3.4 SUMMARY AND DISCUSSION

In this section, we summarize the methods previously discussed based on our taxonomy. We briefly compare them in terms of the current methodologies for table retrieval, features used for this task, experimental datasets for evaluation and relevant approaches for literature benchmarks, which we summarize in Tables 3, 4, 5 and 6.

Table 3 outlines our survey on the core studies for table retrieval (note we sort the articles by methodology). In summary, there is an increasing tendency to utilizing neural models for this task in the last years (CHEN et al., 2020b; SHRAGA et al., 2020c; SUN et al., 2019; TRABELSI; DAVISON; HEFLIN, 2019; ZHANG; ZHANG; BALOG, 2019). Another observation is that the surrounding table context is used in almost all table retrieval work, i.e., document fields such as page title or page abstract (column *Surround Features* in Table 3). In addition, by comparing the single and multi-field methodologies, we note that the re-

cent approaches mostly encode tables as a multi-field (text) document (studies published between 2017 and 2020). Lastly, we observe that several studies apply a combination of distinct techniques for this task in a similar way. For instance, Chakrabarti et al. (2020) consider surrounding table features, external document aspects and entity modeling at once, similar to Venetis et al. (2011). Based on that, we argue that downstream applications for table retrieval can benefit from the combination of relevant findings from the ongoing studies to create a high-quality solution for this task.

We also compare the studies in terms of the query and table features for this task. We go beyond those features as shown in Table 2 (i.e., popular features for regression models in table retrieval) and cover common retrieval aspects for the whole survey, summarized in Table 4. In this direction, we make the following observations. There is a huge set of query, table and document aspects used to the problem of table retrieval. We divide them in two groups: internal aspects and external aspects. The internal aspects cover table features such as table caption, header or body, and other ones come from the matching between query text and table data, i.e., query-dependent features. For instance, some approaches consider the number of times a query term appears over the table data (e.g., $\#hits$ on table headings or table-cells) (CAFARELLA et al., 2008b; CHAKRABARTI et al., 2020). The second group of features is related to document quality measures, i.e., those aspects that do not depend on the query. In addition, other studies have also been used surrounding table features like document title or abstract (i.e., document fields) Gao and Callan (2017). We argue that the content around the table also provides high-quality contextual data to the query/table match.

Besides, we also note that the table caption, headers and body are the most common features for table retrieval. In addition, when we consider human-curated data, the approaches mainly employ external document measures such as page ranking, document citation and page views, beyond query-table term overlaps (BHAGAVATULA; NORASET; DOWNEY, 2013; CAFARELLA et al., 2008b; CHAKRABARTI et al., 2020; PIMPLIKAR; SARAWAGI, 2012; VENETIS et al., 2011). Some solutions also apply non-semantic features into the models including $\#rows$ and $\#columns$ (e.g., (CAFARELLA et al., 2008b)). Lastly, the document title is widely used both in older and recent models (LIU et al., 2007a; SHRAGA et al., 2020a), and the table section title also appears for some work, i.e., surrounding table features (CHEN et al., 2020b; SHRAGA et al., 2020c; ZHANG; BALOG, 2018; ZHANG; ZHANG; BALOG, 2019).

Benchmarks. We also analyze the most common experimental benchmarks for this task. One of the main challenges for the table retrieval task is to construct representative datasets to assess the performance of the devised solutions. *WikiTables* is one of the the most popular benchmarks in the literature: a set of 60 ad-hoc queries and 1.6M tables extracted from Wikipedia created by human relevance judgment (ZHANG; BALOG, 2018). Other public experimental sets include: *WebQueryTable* (SUN et al., 2019), an open-

Table 4 – An overview of the most common table features used by the approaches in the whole survey for table retrieval. We classify the studies in query/table features, document fields as well as document measures for this task.

Reference	**Query/Table Features**									**Doc. Fields**					**Doc. Measures**					
	Table Caption	Table Headers	Table Body	#Table Rows	#Table Columns	#Null Values	#Hits on Headers	#Hits on Body	Table Footnotes	Title	Abstract	Anchor Text	Section Title	Section Index	#Citation	Freshness	Page Rank	#In/Out Links	Page Views	#Hits SurrText
Pyreddy and Croft (1997)	*	*	*																	
Liu et al. (2007a)	*	*	*							*	*				*	*				
Cafarella et al. (2008b)				*	*	*	*	*									*			
Venetis et al. (2011)								*	*			*					*			*
Pimplikar and Sarawagi (2012)		*	*	*	*	*	*	*									*			
Bhagavatula, Noraset and Downey (2013)				*	*	*		*						*			*	*	*	*
Sun et al. (2016)	*	*	*																	
Gao and Callan (2017)	*	*	*						*	*	*									
Zhang and Balog (2018)	*	*	*							*			*							
Sun et al. (2019)	*	*	*																	
Zhang, Zhang and Balog (2019)	*	*	*							*			*							
Trabelsi, Davison and Heflin (2019)	*	*	*							*										
Chakrabarti et al. (2020)			*	*	*	*	*	*									*			*
Chen et al. (2020b)	*	*	*							*			*							
Shraga et al. (2020a)	*	*	*							*										
Shraga et al. (2020c)	*	*	*							*			*							

Source: Created by the author

Table 5 – The most popular table retrieval experimental benchmarks for this task. Seven studies have been used *WikiTables* for evaluation setups.

Corpus	**Reference**
Bing Logs	Chakrabarti et al. (2020)
Google Documents	Cafarella et al. (2008b)
GNQtables	Shraga et al. (2020c)
Marketplaces	Pimplikar and Sarawagi (2012), Sun et al. (2019)
Scientific Documents	Gao and Callan (2017), Liu et al. (2007a), Liu et al. (2007b)
Wall Street Database	Pyreddy and Croft (1997)
WikiTables	Chen et al. (2020), Shraga et al. (2020a), Shraga et al. (2020c), Shraga et al. (2020b), Trabelsi et al. (2019), Zhang et al. (2019), Zhang et al. (2018)

Source: Created by the author

domain dataset consisting of query-table pairs based on search logs from marketplaces; and *GNQtables* (SHRAGA et al., 2020c), a google natural question dataset build on Wikipedia articles. In addition, some studies have evaluated their approaches in particular table domains as scientific documents (LIU et al., 2007a). Table 5 outlines popular experimental datasets for table retrieval in literature.

Table 6 – The best table retrieval results reported by representative methods for this task. We compare the approaches in terms of the NDCG. The symbol (*) means not described in the paper, and bold values are the highest NDCG results (note that we sort the values by NDCG@5).

Reference	NDCG@5	NDCG@10	NDCG@20
(TRABELSI; DAVISON; HEFLIN, 2019)	0.5088	0.5117	0.5587
(ZHANG; ZHANG; BALOG, 2019)	*	0.6096	0.6505
(ZHANG; BALOG, 2018)	0.5951	0.6293	0.6825
(CHEN et al., 2020b)	0.6361	0.6519	0.6564
(SHRAGA et al., 2020a)	0.6498	0.6479	0.6935
(SHRAGA et al., 2020c)	**0.6631**	**0.6813**	**0.7370**
(SHRAGA et al., 2020b)	**0.6654**	**0.6657**	**0.7054**

Source: Created by the author

Relevant Solutions. We conclude our analysis by comparing representative solutions for this task according to our taxonomy in terms of NDCG at cutoffs $k \in \{5, 10, 20\}$. As aforementioned, seven studies have been using *WikiTables* benchmark for evaluation. Based on that, we outline the best table retrieval results for this dataset reported by such studies in Table 6. In summary, the proposed approaches are highly relevant to the task because most of the NDCG scores are very close to each other (as e.g., for the values ranging from 0.5951 to 0.6654 in terms of NDCG@5, an improvement of 0.07 for the best model). Furthermore, the solutions presented by Shraga et al. (2020c) and Shraga et al. (2020b) stand for the best approaches in this dataset, since they outperform the other ones for all evaluated metrics.

3.5 OPEN CHALLENGES

There exist several open challenges and future work that need to be covered in order to produce more representative solutions for this task. We discus some of them as follows.

Evaluation Datasets. Although there are several studies for this task, a small number of evaluation datasets are available. *WikiTables* is the most common benchmark for table retrieval analysis, but such set only covers a very small amount of query-table matches, i.e., only 60 queries are human labeled. Given this gap, few articles have also been shared their experimental datasets for additional evaluations (SHRAGA et al., 2020c; SUN et al., 2019). Towards this front, we argue that further public sets are necessary to assess the performance of the proposed approaches for this task, and therefore downstream studies can also focus on tasks related to the crawler of query-table pairs as well as to the label of representative matches for this task.

Comprehensive Experimental Analyses. As we show in the proposed taxonomy, many solutions are devised for evaluating the matching of unstructured queries and structured web tables. However, to the best of our knowledge, no previous study has evaluated

this pool of approaches for table retrieval over the same experimental environment, i.e., by considering identical configurations of parameter tuning or evaluation benchmarks.

Feature Ablation and Mismatch of Query/Table Terms. Some of the suggested strategies for this task have been using regression models to produce a similarity score for a query-table pair (BHAGAVATULA; NORASET; DOWNEY, 2013; CAFARELLA et al., 2008b; CHAKRABARTI et al., 2020) and non-semantic features including $\#rows$, $\#collumns$ and $\#nullvalues$. None of these studies has proposed an ablation study over the efficacy of such table retrieval features for this task. Further analysis are needed in this scenario. In addition, these studies are mostly based on features that compute the number of overlapping terms between query and table data, e.g., the total of times that a query term appears over the table body. For some type of intent queries, however, there might be a large portion of mismatching information between query and table, which limits the efficacy of these solutions for this task.

Table Cell for Question-Answering. For specific queries, a single row of the table might answer the question. However, only a small number of studies has targeted the task of table cells for QA (CHAKRABARTI et al., 2020; SUN et al., 2016), i.e., beyond retrieving the related tables for a query, also discovering which particular table-row answers it. Further analysis are required in this co-related domain problem.

3.6 CONCLUDING REMARKS

This chapter presented a survey on 16 representative studies for table retrieval in the context of QA. We used a snowballing methodology for the literature review by considering the study of Zhang and Balog (2018) as the start set. Also, we summarize approaches published up to the year 2020. Specifically, our contribution in this chapter was a novel table retrieval taxonomy where the devised solutions for this task were classified into five fronts: probabilistic approaches, traditional IR methods, entity-based models, feature-based techniques and advanced neural networks architectures. Since our survey dates from 2020, we do not include, for instance, transformer-based models in our taxonomy. However, such approaches can also be considered in further investigations. For example, Trabelsi et al. (2022a) propose StruBERT: a model that combines textual and structural information of a table using BERT, also producing context-aware representations for both textual and tabular data. On the same front, other recent studies also include the structured information of the tables for table-matching. Liu et al. (2023) leverage the semantic interaction between table cells and their contexts, in which a dual graph is constructed for row-view and column-view, later aggregated for extracting row and column-oriented features.

Moreover, in this chapter, in addition to covering a set of open challenges for this task including evaluation datasets, ablation studies on table features and more experimental analysis, we also introduced a comparative study in which we analyze the current

methodologies for table retrieval, reported results for evaluation benchmarks and the most common query/tables features for this task (we organized the tables features in query dependent and query independent aspects as well as document fields and document measures). In this thesis work, we address some open challenges correlated to the public evaluation datasets, comprehensive experimental analyses and ablation studies on table retrieval features. First, we present a novel news-table matching corpus for experimentation. By crawling Wikipedia pages, we collected 275,352 news articles and 298,792 web tables for the news-table matching task. Moreover, we perform an extensive experimental evaluation using a set of IR models, in addition to examining which table features are more relevant to the table matching task. Finally, our taxonomy summarizes the most relevant publications for the task of table retrieval in the context of question-answering.

Part II

News Table Matching

4 MATCHING NEWS ARTICLES AND WIKIPEDIA TABLES FOR NEWS AUGMENTATION

Nowadays, digital-news understanding is often overwhelmed by the deluge of online information. One approach to cover this gap is to outline the news story by highlighting the most relevant facts. For example, recent studies summarize news articles by generating representative headlines. In this chapter, we go beyond and argue news understanding can also be enhanced by surfacing contextual data relevant to the article, such as structured Web Tables. Specifically, our goal is to match news articles and web tables for news augmentation. For that, we introduce a novel BERT-based attention model to compute this matching degree. In addition, through an extensive experimental evaluation over Wikipedia tables, we compare the performance of our model with standard IR techniques, document/sentence encoders and neural IR models for this task. In summary, the overall results point out our model outperforms all baselines at different levels of accuracy and in the mean reciprocal ranking measure. As follows, we start this chapter by presenting the news-table matching background.

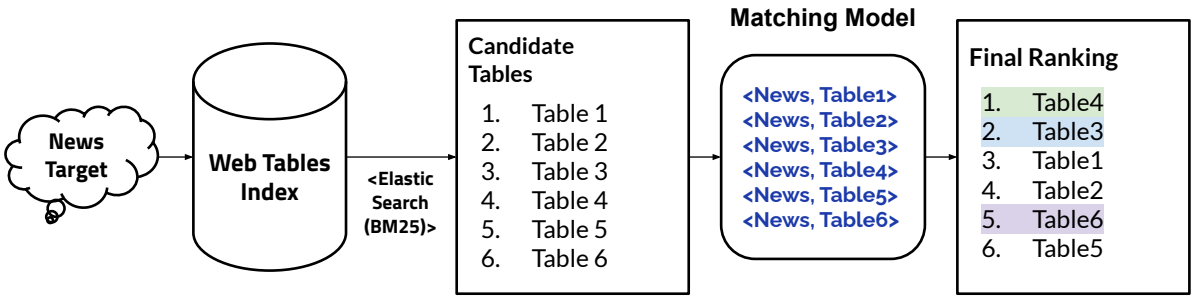
4.1 BACKGROUND

This section covers the news-table matching background. We first formalize this task as a novel rank task. Then, we detail the news-table aspects.

Problem Statement. In this chapter we target the task of matching news articles (A) and web tables (T) as follows: given an article a_i and a set of web tables $T = \{t_1, t_2, t_3, ..., t_n\}$, the goal is to find relevant tables t_i to a_i. Note that the notion of relevance is broad, since a web table can overview news stories, bring contextual data to the topic or answer upcoming questions. Formally, we assume *News-Table matching* as a ranking task, i. e., our goal is to learn a scoring function $f : a_i \times t_j \mapsto \mathbb{R}$ that scores tables in T for a given article a_i to rank them based on news-table aspects.

News-Table Aspects. We represent a news article by three aspects: *title*, *main passage* and *keywords*. The title expresses the central topic of the story. Instead of using the entire news content, we consider the main passage as it compiles the story in a few sentences. For this feature, we collect the meta-description tag from HTML page. Moreover, the keywords represent the most frequent words from the article. Regarding the web tables, which are contained in web pages, we consider the following aspects: *page title* (HTML title), *page main passage* (article's short description), *page keywords* (most frequent words in the article), *table caption, table headers, table body*. The first three ones describe information around the table (i.e., surrounding text) and have been widely used in table retrieval approaches (SUN et al., 2019; LIU et al., 2007b; PIMPLIKAR; SARAWAGI, 2012; VENETIS et al., 2011; ZHANG; BALOG, 2018; DENG; ZHANG; BALOG, 2019). The header indicates the

Figure 15 – News-table matching pipeline. Given a news target and a web table index, we use BM25 to retrieve a set of candidate tables from the table corpus. In sequence, the proposed matching model is applied to this subset to produce the final ranking for the news. Note that *Table4*, *Table3* and *Table6* have shifted their position after the re-ranking step.



Source: Created by the author

properties of the column and helps to describe its meaning. The body (i.e., cells) contains all the table content, and the caption describes the subject of the table. In summary, our matching solution is linked to both the HTML pages and HTML tables, since we also consider the surrounding text of the tables, i.e., the HTML page sections, in addition to extracting the HTML table from the Wikipedia pages. Besides, both article and table aspects are represented by words in natural language, and some table cells can contain numerical values. Lastly, like previous studies (BHAGAVATULA; NORASET; DOWNEY, 2013; SUN et al., 2016; TRABELSI; DAVISON; HEFLIN, 2019; SHRAGA et al., 2020a), we also focus on Wikipedia tables since they are rich corpus of relational information.

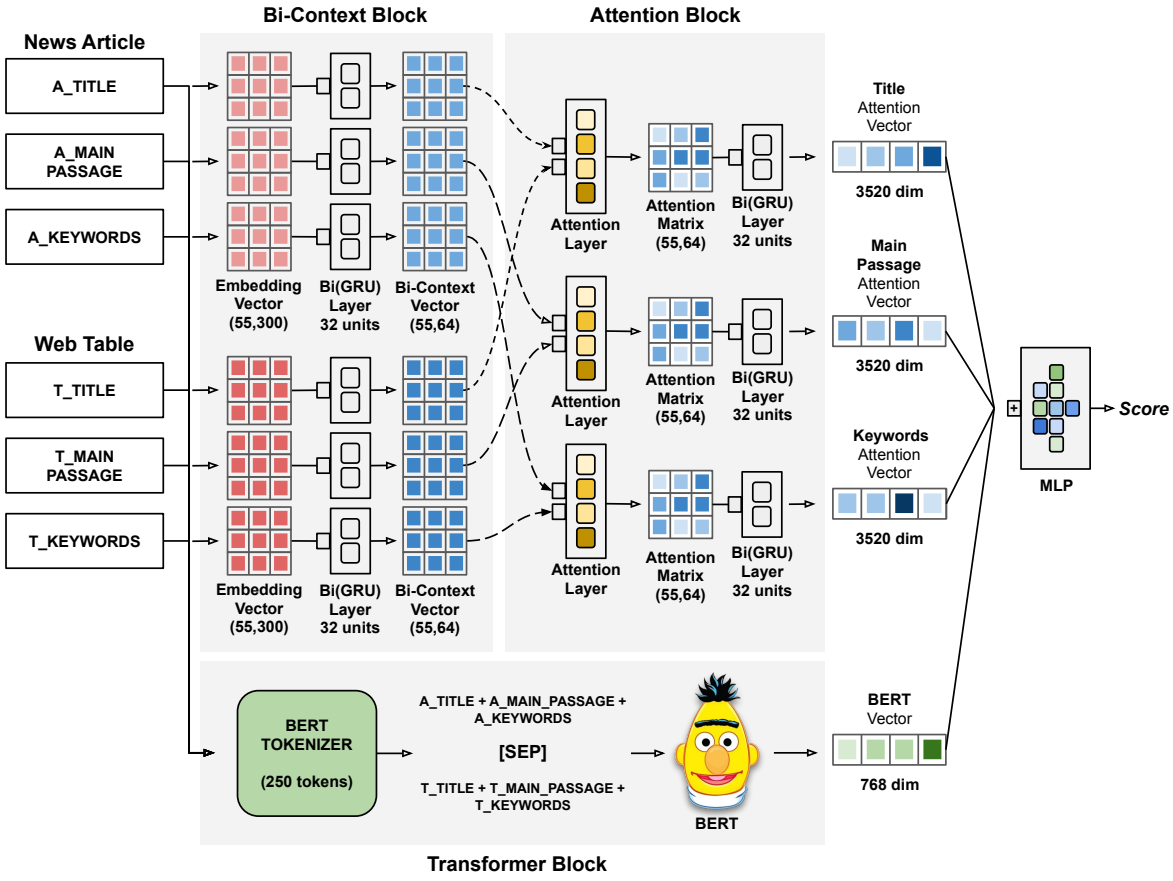
4.2 NEWS-TABLE MATCHING PIPELINE

This section presents our end-to-end solution for matching news articles and web tables, which we illustrate in Figure 15. Our solution has two cascaded steps. First, similar to previous work (SHRAGA et al., 2020a; SHRAGA et al., 2020c; SHRAGA et al., 2020b; SUN et al., 2019), we retrieve a set of candidate tables by using a standard Information Retrieval approach, whose goal is to efficiently find the highest number of relevant tables for the matching model. Next, we use the proposed model to re-rank the candidates in order to obtain the best matching tables.

4.3 NEWS-TABLE MATCHING MODEL

In this section, we introduce the *News-Table* matching model. The core of our contribution is a novel cross-encoder model for this task that performs full attention over the input pair by combining RNN, attention layers and BERT architecture. As a result, we introduce

Figure 16 – An overview of the News-Table Matching Model. Our model learns a joint-representation for a ⟨*news*, *table*⟩ tuple by applying three network blocks: Bi-Context, Attention and Transformer, in which *embedding vector* is the FastText representation from a pre-trained corpus, *bi-context vector* is the contextual vectors learned from the input data, *attention matrix* is the matching degree between news and table aspects and *attention vectors* is the final matching signals for each pair of inputs. Moreover, An MLP architecture captures relevant matches and produces the similarity score on its top layer.



Source: Created by the author

a new way of applying existing neural blocks in the context of news-table matching. In contrast to Lees et al. (2021), which uses BERT as a bi-encoder method, our model learns a joint-representation from a ⟨*news*, *table*⟩ tuple and predicts a similarity score to rank the tables. Our ablation study demonstrates that we obtain better results for table retrieval by merging these blocks in a single network (see results in Table 14). We present the proposed model in Figure 16. It produces two types of representations of the ⟨*news*, *table*⟩ input: one based on cross-attention and another based on BERT. Our goal is to capture relevant matching signals from both sides of the input by using different attention mechanisms. For the attention branch, the input is the embeddings of the words present in the news and table aspects. The network then applies a bidirectional recurrent network

(bi-context block) on them to produce contextual vectors. These vectors are passed to the attention block that combines the aspects of the news article and the table, and outputs an attention vector for each one of them. In the other network's branch, we utilize BERT to obtain another type of contextual representation based on self-attention from the $\langle news, table \rangle$ input. The outputs of two branches are concatenated and passed to an MLP, which computes the matching score on its top layer. The whole network is trained using backpropagation and a binary cross-entropy loss function. We provide details of the model in the remaining of this section.

4.3.1 Input Data Representation

As aforementioned, the model's input is a $\langle news, table \rangle$ tuple. From the news side, we consider the aspects: *article_title*, *article_main_passage* and *article_keywords*. Regarding the table, unlike traditional table retrieval where the answer is inside the table (i.e., headers, caption and body), the most relevant information for *News-Table* matching are those aspects around the table (i.e., its surrounding text): *table_page_title*, *table_page_main_passage* and *table_page_keywords*. We verify this by running previous experiments using a standard retrieval approach whose table content obtained the worst results for this task (see results in Table 9). Lastly, we represent each of those aspects as a sequence of words and utilize a word embedding approach to get word vectors, similar to previous studies (DAI et al., 2018; HU et al., 2015; PANG et al., 2016). The goal here is to generate a dense representation for each news/table token (*Embedding Vector*).

4.3.2 Bidirectional Context

In the context of our task, Recurrent Neural Networks (RNN) have been applied both in IR tasks as well as in table retrieval approaches (SUN et al., 2019; WAN et al., 2016). Overall, these neural models learn contextual information from the sequential data. Since both sides of our input contain consecutive tokens (i.e., article and table aspects are defined by a sequence of words in natural language), we apply RNN to both to get their semantic connections. The goal here is to produce a new representation for the initial embeddings based on the context of the words. In our network, we apply a Gated Recurrent Unit (GRU) (CHUNG et al., 2014) to map each word to a fixed-length vector.[1] As a result, our model learns long text dependencies from each news/table input. Lastly, we also consider a bidirectional GRU to obtain the representation of each word from both directions, and use the concatenation of each hidden state as the final word representation, i.e., the fixed-length vector $= [\overrightarrow{h_{t-1}}, \overleftarrow{h_{t-1}}]$. By doing this, our network learns bidirectional contextual information of each news/table aspect (*Bi-Context Vector*), in order to be sensitive to word order such as reversing or shuffling.

[1] We also try Long Short Term Memory (LSTM) for this step but GRUs achieved better results

4.3.3 Attention

The core challenge of this task is how to compute the similarity degree between distinct news/table features. Overall, news stories are described by several aspects including title, main passage, headlines, keywords and so on. Our intuition is that we can get relevant matching signals from both sides of the inputs by using different attention mechanisms in our network architecture. Based on that, inspired by previous approaches that try to capture relevant association between query/document (HU et al., 2015; LI et al., 2020; MITRA; DIAZ; CRASWELL, 2017; PANG et al., 2016), our model applies attention networks to learn interaction features between article and table, i.e., we use a cross-attention methodology for this network block. Attention mechanisms have been applied in many similar tasks including Question Answering and Text Matching (XIONG; ZHONG; SOCHER, 2017; ZHU et al., 2019). In the context of our task, a matching model needs to identify significant correlation between table aspects and news features such as title, main passage and keywords, in order to capture the best matching information. For that, we use the scaled dot-product attention to weigh news aspects based on table aspects (VASWANI et al., 2017). Specifically, we compute the attention between each co-related aspect: title, main passage and keywords, according to Equation 4.1.

$$Attention(Q, K, V) = softmax(\frac{QK^T}{\sqrt{d_K}})V \tag{4.1}$$

where Q corresponds to the query, K is the key, V the value and $softmax$ is a normalization function. For the title aspect, for instance, we consider *article_title* as Q and V, and *table_page_title* as K. By doing this, we create a novel representation for *article_title* weighted by *table_page_title* (*Attention Matrix*). The other aspects go similarly. In summary, for each pair of them, we generate one attention matrix which represents the matching degree between article and table features. Lastly, the attention matrices fed a Bi-GRU, whose output is flattened, producing the final bi-vectors (*Attention Vectors*).

4.3.4 Transformer

In addition to the attention block, which computes relevant matching for correlated attributes (e.g., article title and table title), we also employ BERT to capture significant interactions between them.[2] BERT is a novel bidirectional sentence encoder based on transformer blocks and multi-head attention mechanisms (SUN et al., 2019). It contains multiple attention heads attending to distinct parts of the sequence at the same time (e.g. longer-term dependencies versus shorter ones). As a result, BERT produces a different representation for each word according to the text sequence where it appears. For example, if we consider the news story in Figure 2 that lists rare Da Vinci paintings (Chapter 1),

[2] Our ablation study shows we can improve the model performance by joining such two attention methodologies (see results in Table 14)

the word *Vinci* has a vector representation when it appears in the news title and another one when it is in the news description since its neighboring words are distinct. Based on that, similar to a previous study that employs BERT for the table retrieval (SUN et al., 2019), we apply it to create a contextual vector representation of the $\langle news, table \rangle$ pair. To fine-tune BERT in our experimental setup, we use the task of sentence pair classification, i.e., a pair of text sentences is classified as match or non-match. For that, we assume the news-table aspects as a single-field (text) document and apply BERT tokenizer to generate $input\_ids$, $attention\_masks$ and $token\_types$ for them. In addition, the token $[SEP]$ separates news-segments from table-segments. Lastly, we adopt the final hidden state h of the first token $[CLS]$ as the whole $\langle news, table \rangle$ vector representation, similar to Sun et al. (2019) (*BERT Vector*).

4.3.5 MLP

On the top of our network, the attention vectors produced by the attention block are concatenated with the BERT vector and fed a Multi-Layer Perceptron Architecture (MLP) to learn matching features and predict a similarity score. The goal here is to capture relevant match signals from both learned vectors. Moreover, the *News-Table* matching score is generated by a sigmoid function over the last MLP neuron. We use this score to rank the matching tables. Lastly, we use the binary cross-entropy loss function for training the whole neural network according to Equation 4.2.

$$Loss = \frac{1}{n} \sum_{i=1}^{n} y_i \; . \; log \; \hat{y}_i + (1 - y_i) \; . \; log(1 - \; \hat{y}_i) \tag{4.2}$$

where $\hat{y}_i$ is the $i-th$ value in the model output, y_i is the corresponding target value, and n is the number of values in the model output.

4.4 EXPERIMENTAL SETUP

In this section, we cover the experimental setup of our evaluation. We present baselines, datasets, methodology, evaluation measures and implementations details.

4.4.1 Baselines

As suggested in previous studies for table retrieval (LEES et al., 2021; SHRAGA et al., 2020c; TRABELSI et al., 2022b), we compare our solution to several state-of-the-art baselines including traditional IR methods, document/sentence encoders, neural IR models and dense passage retrieval strategies.

Traditional IR methods. One can think of the news article as a (long) keyword query and therefore apply traditional IR techniques or QA solutions. We use two different strategies to represent the inputs: news article and Web table. In the multi-field approach,

similar to Zhang and Balog (2018), we compute the similarity score over each aspect separately (e.g., *news title* and *table page title*). Then, we rank the tables based on the mean of similarity/score from all aspects. In the single-field strategy, following Cafarella et al. (2018), we represent both news and table aspects as a single-field (text) document by concatenating their aspects.

- **Cos(TF-IDF)** (SALTON; YANG, 1973). It is a basic IR method that represents query and document terms based on term-frequency and inverse document frequency. It ranks the tables according to the cosine similarity over the TF-IDF vector.
- **BM25** (ROBERTSON; ZARAGOZA, 2009). It is an traditional IR algorithm based on the probabilistic relevance framework that uses term-frequency weighting and document length for ranking documents in a retrieval task.

Document/Sentence Encoders. We evaluate pre-trained sentence/document encoders to represent the news-table aspects. We consider both single and multi-field document approaches, and the cosine similarity is used to score and rank the tables.

- **Doc2Vec** (LE; MIKOLOV, 2014). It is an unsupervised approach that encodes sentences, paragraphs or documents by using neural networks, similar to Word2Vec.
- **USE** (CER et al., 2018). It is a transformer-based network that learns sentence embeddings by using attention. The model was pre-trained on similarity-related tasks such as textual entailment and question/answering.
- **Public-BERT** (DEVLIN et al., 2019). It is a sentence encoder that uses bidirectional transformer networks for language representations. This model was pre-trained on a large corpus from the Internet.
- **Fine-tuned BERT.** We fine-tune BERT to the task of sentence pair classification. For that, we concatenate both news and table aspects as a single-field text document.
- **Lees et al. (2021)**. Given its reproducibility issues, as both NewsBERT and their training data are not publicly available, we implement their network architecture over our news-table corpus. More specifically, we fine-tune BERT by using a pairwise hing loss function over cosine distance, whose goal is to learn that negative (article, table) pairs should have lower similarity than positive pairs.[3] The input for the model is a triple composed of a news article, a positive table and a negative table. They share the same set of parameters for the BERT-encoder.[4]

[3] https://tinyurl.com/ranking-loss

[4] We try the following similarity thresholds for the cosine distance over positive and negative pairs: 0.3, 0.4, 0.5, 0.6, and 0.7, in which 0.3 achieves the best results in our validation dataset

Neural IR Approaches. We also evaluate a set of widely used text matching neural models for IR, which have been applied for downstream tasks including Question Answering (WAN et al., 2016), Paraphrase Identification (PANG et al., 2016), Ad-hoc Search (XIONG et al., 2017) and Document Retrieval (GUO et al., 2016). We use the public Matchzoo Toolkit to train each approach (GUO et al., 2019). In addition, we represent the news-table aspects as a single-field (text) document. In sequence, we use the matching score produced by the respective neural model to rank the tables.

- **DSSM** (HUANG et al., 2013). It maps query and document terms to a low-dimensional space by using TF-IDF, N-Grams and non-linear network layers, and the cosine distance is the query/document score relevance.
- **ARCI** (HU et al., 2015). It is a siamese model that learns semantic representations by using 1D-convolution and max-pooling layers.
- **ARCII** (HU et al., 2015). Unlike previous model, it first builds an interaction space between the two sentences. In sequence, it applies 2D-convolution and max-pooling layers to encode high-level representations.
- **MVLSTM** (WAN et al., 2016). It applies LSTM layers to learn sentence representations for the text input, and a max-pooling interaction step extracts relevant match features from both sentences.
- **DRMM** (GUO et al., 2016). It extracts matches from sentences by word histogram, feed-forward and term gating networks.
- **MATCH PYRAMID** (PANG et al., 2016). It is network-architecture based on image recognition models which learns sentence similarities by using a matching matrix, convolutions and dynamic pooling layers.
- **KNRM** (XIONG et al., 2017). It uses an RBF kernel pooling to learn query-document features from a translation matrix.
- **CONV-KNRM** (DAI et al., 2018). Unlike previous model, it generates a continuous vector for query/document terms by using word embeddings. In sequence, convolutional layers construct n-gram representations of the text. Last, a K-Gaussian kernel pooling counts soft-matches and generates the match score.
- **DUET** (MITRA; DIAZ; CRASWELL, 2017). It is a deep learning model for matching queries and documents by using local and distributed representations of text.

Dense Passage Retrieval Strategies. We complete our baselines by considering novel dense passage retrieval methods. We generate dense representations for both news-table

aspects (as a single-field (text) document). Then, we compute their similarity by applying cosine distance for SBERT and DistilBERT or dot-product for DPR.

- **Dense Passage Retrieval (DPR)** (KARPUKHIN et al., 2020). It is a two independent BERT-based encoder for embedding text sentences, in which the dot-product similarity is used as a ranking function for retrieval.
- **Sentence BERT (SBERT)** (REIMERS; GUREVYCH, 2019). It is a BERT-based model that uses siamese and triplet networks to derive semantically embeddings that can be compared by utilizing cosine-similarity. We also evaluate SBERT by using DistilBERT: a smaller, faster and lighter version of BERT (SANH et al., 2019).

4.4.2 Datasets

This section presents the experimental datasets. We first detail the train/validation corpus. Then, we describe the test data.

Train-Validation Data. Since we are not aware of any public labeled data for this task, we implemented a distant supervision strategy to build a news-table matching dataset. For that, we leverage the links in the reference section on the Wikipedia page that contains the tables. Specifically, we selected only reference links belonging to news web sites. Here, we assume that those news articles are likely related to the table content as they are collocated in the same Wikipedia page. For this evaluation, we gathered 275,352 news articles and 298,792 web tables by adopting Newspaper API.[5] To generate the matching pairs, we index the tables by using a multi-field approach using the Elastic Search (ES) API.[6] Then, for each article, we search over the index by considering its aspects as a single-field query. Lastly, we consider the table with the highest cosine similarity over TF-IDF as the match. By doing this, we collected 93,818 news-table matching pairs that we use for evaluating the proposed model as well as the neural IR approaches. We split this corpus into 84,436 (90%) examples for training and 9,382 (10%) for validating. To the best of our knowledge, this is the first public dataset for the task of matching news articles and web tables in literature.[7] Finally, to demonstrate the reliability of our data, we manually check 100 random samples of matching and non-matching pairs in our train/validation datasets. This evaluation showed that over 92% of them contain correct matching labels. We illustrate a set of them in Table 7.

Test Data. Unlike a common table retrieval task for QA, there is no many public test-sets for matching news articles and web tables. Based on that, since we are not aware of any public labeled test data for this task, we use as a test set a dataset released by Lees et al. (2021). This study is an introductory work to address this task. The authors use an

[5] https://newspaper.readthedocs.io/en/latest
[6] https://www.elastic.co
[7] https://github.com/levysouza/News-Table-Matching

Table 7 – A sample of news-table matching pairs in our training and validating corpus. We show the news-title, the Wikipedia page title for the table and the labels for each one, in which 1 means a matching pair and 0 is a non-match one.

News Title	**Wikipedia Page**	**Label**
Folk Music Awards Nominees Announced	Canadian Folk Music Awards	1
Patty Andrews, Leader Of The Andrews Sisters, Dies	Andrews Sisters	1
Facts About Mexico's Education System	Education in Mexico	1
One Love by David Guetta Reviews	(I'll Never Be) Maria Magdalena	0
The Prime Minister's Official Hub	Megan Hilty	0
Fire Interviews Fraser	Filmfare Award for Best Actress	0

Source: Created by the author

internal production crowd evaluation platform to construct a public dataset comprising 148 news-table pairs created by human labelers. Each pair is labeled by 3 to 5 labelers, and labels are obtained from the majority of ratings. The relevance judgment determines the quality of tables paired to news articles and whether the table provides additional context for, or insight into, the article. In addition to answer whether the table is relevant to the article ("Yes", "So-so", "No"), a question about the table's level of clarity is used to ensure high enough table quality to enable assessment of relevance. Finally, to simulate a real scenario of table retrieval, the authors also released a table corpus containing 53K Wikipedia tables. We use both the table corpus and the ground-truth news-table pairs in our experimental evaluation.

Data Preprocessing. We removed pages with empty values for *title*, *main passage* or *keywords*, and also special characters and stopwords from the text. In addition, we padded long/short sentences based on the average of tokens for each aspect. Table 8 shows a statistical analysis of them on the training, validation and test datasets. As one can see, the aspects have at least 37 tokens on average from the news side (i.e., by jointing news title, description and keywords), which limits the application of QA table retrieval solutions for this task since they usually assume few words in the query.

4.4.3 Methodology

Following previous work (SUN et al., 2019; SHRAGA et al., 2020a; SHRAGA et al., 2020c; SHRAGA et al., 2020b), we assume there is a pool of initial candidate tables for re-raking. Hence, similar to Shraga et al. (2020c), we index the table corpus on Elastic Search API by using a multi-field document approach. Moreover, for each news article, we use ES to obtain the top $k = 100$ candidates tables with the highest BM25 score (by using its default parameter setup). The recall at $k = 100$ using this strategy is 0.9122, which means that for 8.78% of the articles, the match table is not present in the 100 table candidates. Finally, we re-rank them by applying each baseline as well as our proposed model.

Table 8 – A statistical overview for each news-table aspect over the train, validation and test dataset. We point out the minimum and maximum values of the tokens, average of words and standard deviation for each of them.

Aspects	*Train/Validation Data*				*Test Data*			
	Min	**Max**	**Mean**	**STD**	**Min**	**Max**	**Mean**	**STD**
News Title	2.0	28.0	**6.6**	2.7	2.0	23.0	**6.6**	2.9
News Main Passage	2.0	931.0	**18.0**	22.0	2.0	327.0	**18.8**	18.4
News Keywords	1.0	22.0	**13.0**	2.0	6.0	20.0	**13.4**	2.1
Table Title	1.0	13.0	**2.9**	1.3	1.0	13.0	**3.7**	1.4
Table Main Passage	1.0	674.0	**78.4**	40.7	1.0	756.0	**68.1**	63.9
Table Keywords	1.0	18.0	**10.4**	1.5	1.0	20.0	**11.3**	2.3

Source: Created by the author

4.4.4 Evaluation Measures

We evaluate our model and baselines by considering a table retrieval re-ranking task. Like previous IR work (WAN et al., 2016; DAI et al., 2018), we employ Mean Reciprocal Rank (MRR) at cutoff $k = 50$ to assess the average position in which a correctly table-answer appears in the ranking, where the first positions are preferred and, therefore, receive higher scores. In addition, we use accuracy@k (a.k.a. top-k accuracy) at cutoffs $k \in \{1, 5, 10, 20, 50\}$ to measure the percentage of news articles in the test set correctly matched to at least one of the top-k ranked tables. Accuracy@k is a metric widely used in the evaluation of Question Answering tasks (SANTOS et al., 2015) and recommendation systems (MAITY et al., 2019), for which, like for our problem, there are one or few relevant results for each query. We also run a paired Wilcoxon test at 95% confidence level to measure the results' significance. Finally, regarding prediction/latency time of each model, we measure the runtime to retrieve the top-100 candidate tables, produce the similarity scores and obtain the top-20 matching tables for each article. Based on that, we compute the average runtime per article in the test set. We replicate this experiment ten times for each baseline as well as for the proposed model and report the mean for this result.

4.4.5 Implementations Details

We implement the proposed model by using Python 3.6 and TensorFlow 2.2.0. To encode the *news-table* aspects, we use a FastText corpus with 1 million word vectors trained on English Wikipedia pages.[8] Regarding the IR methods, we use TfidfVectorizer from Sklearn for TF-IDF,[9] and Rank-BM25 API for BM25.[10] In relation to the document/sentence encoders, we consider a pre-trained Gensim model for DOC2VEC.[11] Moreover, we import

[8] https://fasttext.cc/
[9] https://scikit-learn.org
[10] https://pypi.org/project/rank-bm25
[11] https://radimrehurek.com/gensim/auto_examples/tutorials/run_doc2vec_lee.html

the 4th version of Universal Sentence Encoder (USE) from TensorFlowHUB.[12] For public BERT, we adopt a online version of Bert-as-Service,[13] and consider TFBertModel from Hugging Face to the fine-turning task.[14] Concerning the neural IR models, we use the Matchzoo Toolkit to train each of them.[15] For dense passage retrieval methods, we use Sentence Transformer API.[16] Finally, we perform the experiments by using a Titan XP GPU and Ubuntu 16.04 LTS.

4.5 RESULTS AND DISCUSSIONS

In this section, we present the news-table matching results in terms of retrieval and ranking. We first discuss the approaches for obtaining candidate tables, i.e., the top-k algorithm. Then, we focus on the models for the ranking step. Moreover, we also introduce an ablation study for prediction time, matching analysis and main components of the proposed BERT-based model.

Top-k Algorithm. The objective of the top-k algorithm is to retrieve the highest number of relevant tables for the matching model (re-ranking). Based on that, we ran a set of distinct index settings to investigate which table aspects obtain the best candidate set, and also evaluate different news features as input queries.[17] We then apply Elastic Search API and BM25 algorithm for retrieval. Such approach has been widely used as a strong baseline for several open-domain retrieval tasks (KARPUKHIN et al., 2020). Table 9 shows the results for each index field and news aspects in terms of *Acc*@100.

Unlike table retrieval for QA, in which the table-content can improve the similarity degree since most answers are inside the table, for news-table matching, the most relevant tables are found by matching the text around the table instead of its content. For example, by combining all news aspects as the input query and the surrounding text of the table as indexes - *page title*, *page main passage* and *page keywords* - the pool of candidates contains over 90% of the ground-truth tables in the set (*Acc*@100 = 0.9122). In contrast, when we use table aspects as indexes like headers or caption, it achieves the lowest results for the same metric. *Table caption* and *table headers* achieve the worst values for *Acc*@100 (only 30% of the matching tables are in the candidate set). Therefore, we do not include these aspects as inputs for the proposed news-table matching model.

Finally, we adopt the best combination of them as the top-k algorithm (*line 20*) and retrieve a pool of candidate tables by querying the index at cutoff $k = 100$. We use this subset to evaluate the baselines and the proposed method over the ranking step as follows.

[12] https://tfhub.dev/google/universal-sentence-encoder/4
[13] https://github.com/hanxiao/bert-as-service
[14] https://huggingface.co/transformers/model_doc/bert.html
[15] https://github.com/NTMC-Community/MatchZoo
[16] https://www.sbert.net/docs/pretrained-models/dpr.html
[17] We do not combine the table-content and its surrounding text since the table-aspects achieve the worst results for the evaluated metric in our experiments

Table 9 – Searching over the index by using Elastic Search API and BM25 algorithm (Top-k Algorithm). We evaluate distinct approaches for news-table aspects. Bold values represent the best combination for the candidate set in terms of *Acc*@100, and the symbol (∗) points out the worst results for the same metric (when we use table aspects as indexes).

#	**News Aspects (*Query*)**	**Table Aspects (*Indexes*)**	**Acc@100**
1	Title	Title	0.6959
2	Title	Main Passage	0.6622
3	Title	Keywords	0.7279
4	Title	*Table Caption**	0.0680
5	Title	*Table Headers**	0.1216
6	Title	*Table Body**	0.3176
7	Main Passage	Title	0.5682
8	Main Passage	Main Passage	0.5000
9	Main Passage	Keywords	0.5676
10	Main Passage	*Table Caption**	0.0621
11	Main Passage	*Table Headers**	0.1284
12	Main Passage	*Table Body**	0.3041
13	Keywords	Title	0.7343
14	Keywords	Main Passage	0.6892
15	Keywords	Keywords	0.7162
16	Keywords	*Table Caption**	0.0816
17	Keywords	*Table Headers**	0.2905
18	Keywords	*Table Body**	0.5608
19	Title, Main Passage	Title, Main Passage	0.7838
20	**Title**, **Main Passage**, **Keywords**	**Title**, **Main Passage**, **Keywords**	**0.9122**

Source: Created by the author

Ranking Results. We now discuss the core results of our study, i.e., the ranking step, which we present in Table 10. We first examine the ranking accuracy of the approaches then we focus on their average prediction time. Given the pool of candidate tables, we re-rank them by applying each baseline as well as our proposed model. Overall, our model outperforms all baselines for all evaluation metrics. In fact, we run a paired Wilcoxon Test at 95% confidence level between the MRR score of our model and each baseline which showed that its MRR value is statistically different than all baselines (See Table 11 for the p-values comparisons). In terms of *ACC*@1, our model correctly ranks over 55% of the ground-truth tables, and at the top-five ranking positions (*ACC*@5), it achieves accuracy of 77%. Comparing its results, for instance, with *multi-field USE* (a pre-trained matching encoder), our model is at least 36% more effective in the re-ranking step. Regarding the *Fine-tuned BERT* (the strongest baseline), our model surpass it by over 13%. In contrast to Lees et al. (2021), our model surpasses their network architecture by a large margin for all evaluated metrics. In fact, previous studies have demonstrated that BERT bi-encoders,

Table 10 – Ranking results for the proposed model and each baseline. We consider the following attributes for matching. News aspects: *Title*, *MainPassage*, *Keywords*. Table aspects: *Page Title*, *Page MainPassage*, *Page Keywords*. The symbol (*) means a statistically significant better result compared to all baselines, (SF) is a Single-Field approach and (MF) is a Multi-Field approach. We also show the average prediction time in seconds for each news article in the test set.

APPROACH	**MRR@50**	**Acc@k=1**	**5**	**10**	**20**	**Avg. time(s)**
Cos(TF-IDF) - (SF)	0.4414	0.3176	0.5946	0.7297	0.7973	0.2777
BM25 - (SF)	0.4269	0.2973	0.5946	0.6959	0.7432	**0.0456**
DOC2VEC - (SF)	0.2734	0.1892	0.3649	0.4662	0.6284	1.2770
USE - (SF)	0.3837	0.2838	0.4932	0.5811	0.6622	0.4078
PUBLIC-BERT - (SF)	0.3313	0.2230	0.4459	0.5541	0.6824	3.1564
Cos(TF-IDF) - (MF)	0.4513	0.3649	0.5405	0.6081	0.7365	0.7624
BM25 - (MF)	0.4069	0.3041	0.5405	0.6351	0.6824	0.0594
DOC2VEC - (MF)	0.4248	0.3041	0.5203	0.6622	0.7500	1.3787
USE - (MF)	0.5166	0.4054	0.6419	0.7297	0.8108	0.4634
PUBLIC-BERT - (MF)	0.3373	0.2432	0.4459	0.5068	0.5811	8.4924
Fine-tuned BERT	0.5949	0.4865	0.7500	0.7905	0.8514	1.9803
Lees et al. (2021)	0.1510	0.0743	0.1824	0.2905	0.5203	1.5733
DSSM	0.0348	0.0068	0.0338	0.0878	0.1959	0.2937
ARCI	0.0768	0.0270	0.0878	0.1554	0.3378	0.3044
ARCII	0.0667	0.0135	0.0946	0.1486	0.3311	0.6722
MVLSTM	0.0765	0.0068	0.1216	0.2432	0.4122	0.3518
DRMM	0.0486	0.0270	0.0473	0.1081	0.2365	6.5425
MATCH PYRAMID	0.0545	0.0068	0.0676	0.0878	0.1689	0.4956
KNRM	0.1285	0.0608	0.1689	0.2703	0.4392	0.4032
CONV-KNRM	0.1103	0.0608	0.1486	0.1959	0.2703	0.5717
DUET	0.1560	0.0608	0.2365	0.3784	0.5405	0.5013
DPR	0.4550	0.3311	0.6419	0.7568	0.8243	1.1242
SBERT	0.5135	0.3851	0.6824	0.7770	0.8514	0.3054
DistilBERT	0.5045	0.3986	0.6216	0.6824	0.8041	0.5081
OUR-METHOD	**0.6369***	**0.5541**	**0.7703**	**0.8176**	**0.8514**	2.2760

Source: Created by the author

used by Lees et al. (2021), usually have lower performance in comparison with BERT cross-encoders (THAKUR et al., 2021), which we apply in our solution. Another possible reason for their poor performance is that in their results NewsBERT attains over 45% of the matching tables for $ACC@1$. But, since NewsBERT is a private Google resource, we implemented their approach using a public BERT in our experimental setup. Lastly, concerning the dense passage retrieval methods such as SBERT and DPR, our solution outperforms them by over 20% in terms of $MRR@50$.

We next analyze the performance of the single and multi-field baselines, neural IR models as well as the DPR approaches. Similar to traditional table retrieval strategies, news-table ranking results are improved by adopting a multi-field methodology. For example, $ACC@1$ improves from 0.1892 to 0.3041 (*DOC2VEC*), 0.2838 to 0.4054 (*USE*) and

Table 11 – P-values results for Wilcoxon Test. We compare our model against each baseline in terms of MRR@50. Our approach statistically outperforms all baselines.

RANKING METHOD	**P-Value**
Cos(TF-IDF) - single-field	$1.5329 * 10^{-08}$
BM25 - single-field	$1.8467 * 10^{-08}$
DOC2VEC - single-field	$6.7410 * 10^{-15}$
USE - single-field	$1.0348 * 10^{-09}$
PUBLIC-BERT - single-field	$3.5424 * 10^{-13}$
Cos(TF-IDF) - multi-field	$1.0992 * 10^{-07}$
BM25 - multi-field	$2.2055 * 10^{-09}$
DOC2VEC - multi-field	$5.0557 * 10^{-08}$
USE - multi-field	0.0001
PUBLIC-BERT - multi-field	$3.0637 * 10^{-14}$
Fine-tuned BERT	0.0077
Lees et al. (2021)	$1.4215 * 10^{-19}$
DSSM	$1.2386 * 10^{-23}$
ARCI	$2.2968 * 10^{-21}$
ARCII	$1.8398 * 10^{-22}$
MVLSTM	$4.9307 * 10^{-22}$
DRMM	$4.8761 * 10^{-22}$
MATCH PYRAMID	$1.0299 * 10^{-22}$
KNRM	$8.2976 * 10^{-20}$
CONV-KNRM	$1.4310 * 10^{-20}$
DUET	$1.1611 * 10^{-19}$
DPR	$9.5979 * 10^{-7}$
SBERT	0.0002
DistilBERT	$4.0489 * 10^{-05}$

Source: Created by the author

0.2230 to 0.2432 (*PUBLIC-BERT*) when a multi-field approach is used. Indeed, each news or table aspect individually contributes to the ranking score. The neural IR models achieve the worst results: their ACC@1 is close to zero. Even when combining other news-table aspects, the neural IR models do not outperform any other ranking method. A possible reason for the poor performance of these models is they were devised to answer short queries and, in the context of our task, most of the queries are long, i.e., the concatenation of the news aspects. Among the neural IR models, DUET obtains the best performance: $ACC@5 = 0.2365$. Finally, concerning dense passage retrieval models, *SBERT* and *Distil-BERT* surpass *DPR* in terms of Acc@1 and MRR@50. Moreover, our study also confirms that these approaches attain better results for table retrieval than traditional IR methods like Cos(TF-IDF) and BM25. For example, *SBERT* supasses Cos(TF-IDF) - (MF) by over 12% in terms of MRR@50.

Table 12 – News-table matching results by adopting a maximum recall scenario in the retrieval set. The symbol (*) means a statistically significant better result compared to the other baselines.

Top-K Algorithm	RE-RANK	MRR@50	Acc@k=1	5	10
BM25	Cos(TF-IDF) - (MF)	0.4530	0.3649	0.5473	0.6149
Cos(TF-IDF)	Cos(TF-IDF) - (MF)	0.4778	0.3919	0.5541	0.6351
BM25	USE - (MF)	0.5345	0.4122	0.6757	0.7703
Cos(TF-IDF)	USE - (MF)	0.4416	0.3176	0.5676	0.6892
BM25	**OUR METHOD**	**0.6488***	**0.5608**	**0.7838**	**0.8514**
Cos(TF-IDF)	**OUR METHOD**	**0.6959***	**0.6216**	**0.8176**	**0.8649**

Source: Created by the author

On the whole, *USE* and *Fine-tuned BERT* are the strongest baseline as they achieve better results than the traditional IR methods, neural models and sentence encoders in terms of *ACC*@1 and MRR. In fact, dense retrieval approaches, which apply BERT as encoder, have shown comprehensive efficacy at several open-domain IR tasks (KARPUKHIN et al., 2020). Our results also confirm such hypothesizes for news-table matching. In addition, the IR methods, although simpler, achieve good results and are strong baselines. For example, in *ACC*@1, Cos(TF-IDF) multi-field surpasses both all neural IR models and sentence encoders such as *DOC2VEC* and *PUBLIC-BERT*.

Lastly, similar to previous work (SHRAGA et al., 2020b), we assume there is a pool of candidate tables in which our model applies re-ranking. As aforementioned, in this evaluation, this candidate pool with $k = 100$ has recall of 0.9122. To evaluate our approach in a 100% recall scenario, we added the correct table to the pools that do not contain it (8.78% of the news articles). Table 12 shows such results. Also in this scenario, our method outperforms the strongest baselines in terms of *ACC*@1 and MRR. In addition, even when we change the the top-k retrieval algorithm to collect the candidate pool (i.e., BM25 to classic TF-IDF), the proposed model outperforms the baselines. That result confirms our method correctly re-ranking the web tables regardless of the retrieval approach. Finally, over a maximum recall scenario, our method ranks 62.16% of the ground-truth tables at the first ranking position using Cos(TF-IDF) as the top-k algorithm.

Prediction Time. We now discuss the query prediction time of each ranking model. We measure the average runtime per news article in the test dataset as shown in Table 10 (over the last column). Overall, the algorithm BM25 for both single and multi-field approaches has the smallest time, 0.0456 and 0.0594 seconds per query respectively, while the models *DRMM* and *PUBLIC-BERT - (MF)* have the longest ones (6.5425 and 8.4924 seconds respectively). In contrast, *PUBLIC-BERT* uses external web services which leads to higher latency. For *DRMM*, the model combines several components including matching histogram mapping, feed forward networks, terms gatting networks and term vector frequencies. As a result, it has a high runtime. Regarding single and multi-field approaches, the time per query is very similar. For example, BM25, *DOC2VEC* and *USE* have sim-

ilar runtimes. In relation to the neural IR models such as *DUET* and *CONV-KNRM*, their execution time per query is close to 0.5 seconds, but they have poor performance in terms of accuracy. For the novel dense passage retrieval techniques, *DPR* is the slowest algorithm (1.1242 seconds). *SBERT* is over three times faster than *DPR*.

Lastly, our cross-encoder model has a prediction time of 2.2760 seconds per news article. Comparing its results, for instance, with *Fine-tuned BERT* (the strongest baseline), our model is over 0.2 seconds slower but over 13% more effective in terms of *ACC*@1. Such results also indicate that the combination of blocks, used in our network, does not put high latency on query prediction. *Bi-GRUs* and *attention layers* increase over 0.2 seconds in the final time compared to the *Fine-tuned BERT*, which only uses the BERT architecture. In contrast to Lees et al. (2021), bi-encoder models are faster than ours but have lower accuracy [18]. Finally, a possible alternative to decrease the runtime of our model is to use a distilled version of BERT instead the original one in the *Transformer Block* since it shows a much smaller runtime.

Matching Analysis. We now present three matching examples for the test set. For each news story, we collect the $top - 5$ tables produced by our model, which we illustrate in Table 13 (note we also show their similarity degree). This results demonstrate our model re-ranks correlated tables for each of them in the first ranking positions. For example, regarding Article 1, which contains facts about *Cars*, *America*, *Chrysler* and *Ford*, our model points out tables such as *Chrysler Vehicles* and *Ford Vehicles*. In fact, a reader of this news article may be interested in knowing which cars are manufactured by the Chrysler/Ford automakers. In addition, our approach also finds matching tables in which there is no term-overlap for the news-table titles, i.e., exacting matching (e.g., *Automobiles Manufactured in United States*). Such linking provides further information about the central topic of the story - *cars made in America* - beyond exploring specific places for this news as the United States and Ontario. Regarding the second article, *NASA's Moonwalking Apollo Astronauts*, the results are similar since our model finds tables like Apollo Missions or Astronauts, Spacewalks and Moonwalks (very relevant tables to this news story). As a result, any reader can further explore the list of all Apollo missions or astronauts which landed on the Moon. Lastly, for Article 3, which relates to Best-Selling Video Games, our approach retrieves matching tables such as *Nintendo* and *Gamecube Video Games*, i.e., specific brands for games.

Ablation Study. We conclude this section by presenting an ablation study of our model. As we show in Figure 16, our network combines three main components: *Bi-Context Block*, which uses recurrent networks to learn contextual vectors from the input; *Attention Block*, whose goal is to compute the matching degree between article and table features; and *Transformer Block*, which applies multi-head attention layers based on BERT architecture.

[18] The training time for cross and bi-encoder BERT-based models are similar as both of them are composed of the BERT architecture. Their fine-tuning time is over 20min per epoch in our experiments

Table 13 – A sample of news-table matching in the test set. We present the $top-5$ tables for three evaluated news articles, beyond pointing out their similarity degree.

Article 1 - Title: *Cars made in America? Chrysler, Ford no longer qualify.*		
#	**Table Title**	**Similarity**
1	Chrysler Vehicles	0.9861
2	Ford Vehicles	0.9104
3	Toyota Vehicles	0.8727
4	Automobiles Manufactured in United States	0.8223
5	Automobiles Manufactured in Ontario	0.8023
Article 2 - Title: *NASA's Moonwalking Apollo Astronauts.*		
#	**Table Title**	**Similarity**
1	Apollo Astronauts	0.9976
2	Apollo Missions	0.9910
3	Missions of the Moon	0.8966
4	Spacewalks and Moonwalks	0.8891
5	Spacewalkers	0.8875
Article 3 - Title: *The Best-Selling Video Games.*		
#	**Table Title**	**Similarity**
1	Best Selling Video Games	0.9476
2	Best Selling Nintendo Video Games	0.9130
3	Best Selling Gamecube Video Games	0.9106
4	Games Gold Games	0.9068
5	The Simpsons Couch Gags	0.9022

Source: Created by the author

Table 14 – Ablation study on the proposed model for its network-blocks. We evaluate the following components and their combinations: Bi-Context Block, Attention Block, Transformer Block and Full Model.

#	**Network Block**	**MRR@50**	**Acc@k=1**	**5**	**10**
1	Bi-Context	0.0913	0.0270	0.1419	0.2568
2	Attention	0.1647	0.0811	0.1959	0.3176
3	Transformer	0.5949	0.4865	0.7500	0.7905
4	Bi-Context + Attention	0.3768	0.3108	0.4324	0.5405
5	Bi-Context + Transformer	0.6236	0.5270	0.7432	0.8108
6	Attention + Transformer	0.6193	0.5135	0.7703	**0.8311**
7	Full Model	**0.6369**	**0.5541**	**0.7703**	0.8176

Source: Created by the author

We evaluate each block individually as well as their combinations. Table 14 shows the results for each of them in terms of $Acc@k$ and $MRR@50$. Overall, if we only use *Bi-Context Block* (*line 1*) or *Attention Block* (*line 2*) for matching, the model achieves the worst results for this task (its $Acc@1$ is equal to 0.0270 and 0.0811 respectively). In contrast, by combining recurrent networks and attention layers (*line 4*), the model finds over 30% of the matching tables at the first rank position (almost four times better than these isolated blocks). Specifically, the *Transformer Block* achieves the best results for

MRR@50 and *Acc*@*k* compared to the other network components (*line 3*). Moreover, if we concatenate it with both recurrent networks or attention layers (*lines 5* and *6*), the results also increase for the same metric. For example, its results improve from 0.4865 to 0.5270 for *Acc*@1 (*line 5*). The results are similar to the combination of *Attention* and *Transformer*. Finally, by analyzing all blocks and their combinations, the *Full Model* attains the highest results for news-table matching (over 55% for *Acc*@1).

Such results indicate our approach increases the performance by over 13% in terms of *Acc*@1 compared to the *Transformer Block* (the most isolated baseline). In addition, we also confirm that recurrent neural networks and cross-attention layers can also capture relevant match signals from the input in a news-table matching task.

4.6 CONCLUDING REMARKS

The task of matching news articles and web tables is a recent table retrieval problem. In this chapter, we claimed news understanding can be enhanced by joining associated content from structured web tables. In fact, previous studies have demonstrated that online readers also explore tables inside Wikipedia pages after looking at news articles. Based on that, we focused on the task of the *news-table* matching. Our solution for that is a hybrid neural network that combines different encoders to better represent articles and tables for this task. Our intuition was that we can improve the similarity degree by using distinct attention approaches in the same network architecture.

We performed an extensive evaluation that assessed the performance of our approach, comparing it with standard IR methods, document/sentence encoders, neural IR models and dense retrievers. Furthermore, we also included an ablation study on the main components of our neural network, in addition to analyzing a set of news-table matching examples produced by our model. In comparison to Lees et al. (2021), the most related baseline, our study provides further directions in the context of *News-Table* matching. Finally, over a set of wikipedia tables, the overall results point out our model outperforms the baselines for all evaluated metrics.

Part III

Dataset Search

5 IMPROVING DENSE RETRIEVAL MODELS WITH LLM AUGMENTED DATA FOR DATASET SEARCH

Data augmentation for training supervised models has achieved great results in different areas. With the popularity of Large Language Models (LLMs), a research area has emerged focused on applying LLMs for text data augmentation. This approach is particularly beneficial for low-resource tasks, whereby the availability of labeled data is very scarce. At this front, dataset search is an information retrieval task that aims to retrieve relevant datasets based on user queries. However, due to the lack of labeled data tailored explicitly for this task, developing accurate retrieval models becomes challenging. In this chapter, we cover this gap by targeting LLMs to create training examples for retrieval models in the dataset search task. Specifically, we propose a new pipeline that generates synthetic queries from dataset descriptions using LLMs. The query-description pairs are utilized to fine-tune dense retrieval approaches for re-ranking, which we assume as soft matches to our task. We evaluate our pipeline using fine-tuned embedding models for semantic search over dataset search benchmarks (NTCIR and ACORDAR). We fine-tune these models in the dataset search task using the synthetic data generated by our solution and compared their performance with the original models. In summary, the overall results show that the models tuned on the synthetic data statistically outperform the baselines at different normalized discounted cumulative gain levels. As follows, we begin this chapter by covering the dataset retrieval background we consider in this thesis work.

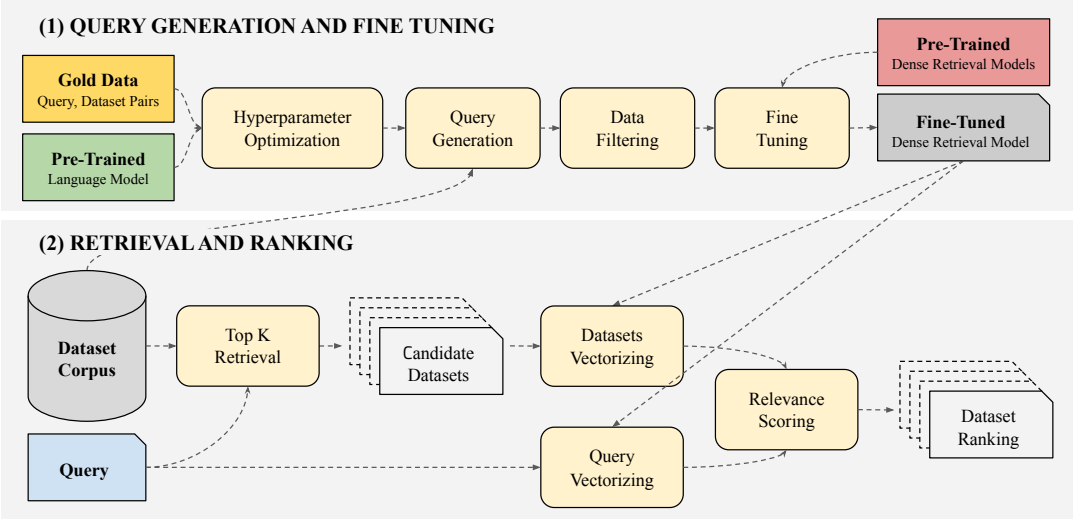
5.1 PROBLEM STATEMENT

This section formalizes the dataset retrieval task. In addition, we detail the data augmentation task for query generation.

Dataset Retrieval Task. In this chapter, we formalize our dataset retrieval task as follows: given a query q and a collection of datasets $D = \{d_1, d_2, d_3, ..., d_n\}$, our goal is to learn a scoring function $f : q \times D \mapsto \mathbb{R}$ that scores datasets in D for q. Similar to previous work (CHEN et al., 2020a), we assume d is the dataset's metadata that comprises its title, description, and keywords: $d = \langle title, description, keywords \rangle$. The title summarizes the core information about the data, the description contains short details regarding its lines and rows, and the keywords represent the main topics of the dataset. Moreover, we also consider that queries for this task are generally short, having between one and three words (KACPRZAK et al., 2017; KACPRZAK et al., 2018; KACPRZAK et al., 2019).

Data Augmentation Task. Given a collection of datasets D and a pre-trained LLM, we aim to generate a set of matching pairs $M = \{\langle d_1^{\oplus}, sq_1 \rangle, \langle d_2^{\oplus}, sq_2 \rangle, ..., \langle d_n^{\oplus}, sq_k \rangle\}$ for each $d_i \in D$, such that $sq_k = LLM(d_i)$ and represents the augmented text query created by the large language model for the dataset d_i.

Figure 17 – DAPDR: our Data Augmentation Pipeline for Dataset Retrieval in two branches: (1) query generation and fine-tuning; (2) retrieval and ranking. We utilize LLMs to produce synthetic queries for each dataset in the corpus. By filtering query-dataset pairs, we apply them to fine-tune dense retrievers for ranking. On the other branch, the top-k candidate datasets are vectorized by applying the fine-tuned models, in addition to measuring the relevance score between queries and datasets for obtaining the final dataset ranking.



Source: Created by the author

5.2 THE DATASET RETRIEVAL PIPELINE

The core of our solution, DAPDR (Dataset Retrieval Pipeline), is to generate synthetic queries of dataset descriptions using pre-trained LLMs to train ranking models for the dataset retrieval task. DAPDR has two branches, which we illustrate in Figure 17: ***(1) query generation and fine-tuning***; and ***(2) retrieval and ranking***. In the first one, DAPDR leverages *docTTTTTquery* to create augmented queries for each dataset, optimizing its hyperparameters over the gold data. DAPDR filters query-dataset pairs by removing mismatching questions using a similarity-based strategy. The filtered corpus is the input to fine-tune dense retrieval models.

For the retrieval and ranking pipeline, DAPDR uses IR methods to retrieve the $top-k$ candidate datasets for each query. It then applies a fine-tuned model to vectorize the datasets' metadata and the query. Finally, DAPDR ranks the datasets based on their cosine similarity to the query.

In contrast to the previous work (CHEN et al., 2020a), our major difference is that we use LLMs to produce soft-matching queries for datasets, later utilized for fine-tuning dense retrievers. We also apply the tuned retrievers on the target task for re-rank. Besides, we are the first study to employ LLMs for the dataset retrieval task. We provide further details for each step of our pipeline in the remainder of this section.

5.2.1 Query Generation and Fine-Tuning

This branch of DAPDR creates query-dataset pairs for fine-tuning dense retrieval models. It includes the following components: hyperparameter optimization; query generation; data filtering; and fine-tuning. We detail each of them in the remainder of this section.

Hyperparameter Optimization. Similar to previous work (BONIFACIO et al., 2022; JERONYMO et al., 2023; NOGUEIRA et al., 2019; SACHAN et al., 2022), we target LLMs in the context of query generation for data augmentation. Instead of using prompting strategies such as one-shot or few-shot learning (BROWN et al., 2020), we consider *docTTTTTquery*:[1] an adapted T5-based model for this task trained on MS MARCO Passage Dataset.[2] T5 is an encoder-decoder model for language modeling (RAFFEL et al., 2020). Built on top of the Transformer network, its core idea is to convert each task domain to a text-to-text format, where an input text feeds the model, which is then asked to produce some text output. In addition, a task-specific (text) prefix is also included in the input sentence, which specifies the target task for the text prediction. For instance, an example of a task prefix is: *translate English to German: That is good.* One motivation for choosing *docTTTTTquery* is that it is an open-source model. Given a text document, it produces relevant queries for the document in a zero-shot way. Furthermore, in contrast to previous approaches that use default values for the LLM's decoder hyperparameters (as, e.g., for *top-k* and *top-p*) (BONIFACIO et al., 2022; NOGUEIRA et al., 2019), we perform a search for their best values for our task (HOLTZMAN et al., 2020). Specifically, for each query-dataset pair in the gold data,[3] we ask *docTTTTTquery* to create k[4] artificial queries. The concatenation of the dataset title and description is the input to *docTTTTTquery*. Moreover, DAPDR uses two different strategies to optimize its hyperparameters. For a given dataset, the first one calculates the mean similarity between the k synthetic queries and a human-curated query for this dataset. The second one selects the max similarity among the k queries. Equation 5.1 and 5.2 describe our similarity strategies for hyperparameter optimization.

$$S = \frac{1}{k}\sum_{i=1}^{k} Cos(q, sq_i) \tag{5.1}$$

$$S = max(Cos(q, sq_1), \cdots, Cos(s, sq_k)) \tag{5.2}$$

where q represents the vector of the human-curated query for the dataset, sq_i is the ith vector for the synthetic one, and k is the number of artificial queries. The goal of the optimization is to maximize the similarity between the gold query and the augmented ones. As a result, we aim that the tuned *docTTTTTquery* produces synthetic queries similar to the human-curated ones. We utilize SBERT as the embedding model for this

[1] http://huggingface.co/BeIR/query-gen-msmarco-t5-large-v1
[2] http://microsoft.github.io/MSMARCO-Passage-Ranking
[3] NTCIR has 141 relevant pairs, and ACODAR has 692
[4] We ask the LLM to create five queries in our experiments

Table 15 – Hyperparameter optimization for *docTTTTTquery*. We evaluate *top-p*, *top-k*, *temperature* and *beams* in this setup. The best values for each one are presented for NTCIR and ACORDAR benchmarks (see Appendix A for more details).

Hyperparameter	**Search**	**NTCIR**	**ACORDAR**
Top-P	[0,1]	0.9881	0.9775
Top-K	[1,50]	22	25
Temperature	[0,1]	0.9989	0.9829
Beams	[1,3]	1	1

Source: Created by the author

Table 16 – A sample of the augmented queries created by *docTTTTTquery*. We present the dataset title and the augmented query for NTCIR and ACORDAR benchmarks. We summarize distinct types of queries.

NTCIR Dataset	
Dataset Title	**Augmented Query**
medicaid financial management data	what states participate in medicaid
oil rig weather observations	where are oil rigs located
hatchie national wildlife refuge land status map	who owns hatchie national wildlife refuge
oasdi beneficiaries state county 2005	oasdi beneficiary population
a state art review effects fire wetland ecosystems	effects of fire in an ecosystem
ACORDAR Dataset	
Dataset Title	**Augmented Query**
electronic waste generated recycled	what is recycled electronic waste
2014 rain gage stations location	where are the rain gage stations located
historical occupational business licenses 2013	who required history of occupational license
2015 solar survey responses	u.s. solar energy survey response
poverty estimates washington counties age	washington state poverty population

Source: Created by the author

step (REIMERS; GUREVYCH, 2019), and Optuna to perform bayesian optimization with 20 trials.[5] Table 15 shows our setup for the optimization step and the best values for each hyperparameter in both benchmarks. As in previous studies (HOLTZMAN et al., 2020), we also achieve the best results in our experiments by using values close to 1.0 for *top-p* and *top-k*, which tends to decrease text repetition. Also, we do not notice any changes by increasing the range for the total of beams.

Query Generation. In this step, DAPDR asks *docTTTTTquery* tuned in the previous step to generate a set of synthetic queries for each dataset in the corpus. We assume them as soft matches for our task, which we use to fine-tune the dense retrieval models for ranking, similar to previous work (BONIFACIO et al., 2022). Specifically, DAPDR creates five queries for each dataset. As human-curated queries for this task are usually short (KACPRZAK et al., 2019), it generates queries from 3 to 10 tokens. Our final match-

[5] https://optuna.org

ing corpus contains over 171k query-dataset pairs for NTCIR and 117k for ACORDAR. Besides, since we adjust *docTTTTTquery* hyperparameters on NTCIR and ACORAR gold data, we also double-check whether some of the synthetic queries produced by the LLM overlap the original queries on ground truth (as such corpus are also used for experimental evaluation). According to this analysis, we confirm that there is no overlapping between the original queries and the artificial queries produced by *docTTTTTquery*, i.e., all augmented queries for this task are different from those on NTCIR and ACORAR gold data. Table 16 shows a sample of the augmented queries for both benchmarks. We observe that *DocTTTTTquery* produces queries related to their respective datasets. For example, the query *Where are Oil Rigs Located* in NTCIR, which searches for oil platforms, maps to the dataset *Oil Rig Weather Observations*, since it includes oil sites in addition to weather. The same applies to the augmented query *U.S. solar energy survey response* in ACORDAR, which matches the data *2015 Solar Survey Responses*.

Data Filtering. The Query Generation step can eventually generate mismatching queries. For instance, the augmented question *What is the Mixed Methods Process Evaluation?* is not a good match for the dataset *Cabo Verde Water Sanitation Hygiene*, neither the query *When was Division Realty?* is for *1954 Annual Lands Report*. Such query examples have been created by the LLM. To deal with that, DAPDR filters the query-dataset pairs by employing a distance supervision approach similar to previous studies (BONIFACIO et al., 2022; JERONYMO et al., 2023). Specifically, it computes a relevance score for each pair by calculating the average similarity between the vector embedding of the generated query and the dataset metadata (title and description). Pairs with similarity below a given threshold are excluded. We empirically selected the best similarity cut-off, as we report in our experimental evaluation. We utilize SBERT as the embedding model for this step, and cosine as the similarity score. After this step, the filtered pairs are used for fine-tuning the ranking models. Table 17 presents the total of training samples for each benchmark according to the similarity threshold.

Fine Tuning. In the context of LLMs for data augmentation, previous studies have applied them for fine-tuning tasks (BONIFACIO et al., 2022; DAI et al., 2022; JERONYMO et al., 2023). On the same front, DAPDR uses the augmented query-dataset pair for fine-tuning retrieval models. Note that *docTTTTTquery* only creates matching pairs. For the negative ones, DAPDR randomly samples datasets from the corpus for each synthetic query, similar to previous work (BONIFACIO et al., 2022). It includes negative pairs in the same proportion as the matching ones. The model outputs a score for the matching or non-matching example by considering the cosine similarity loss (Equation 5.3), in which *label* represents the pair's matching score (0 or 1), *cosine_sim* stands for the cosine similarity, and u and v is the embedding vector for query and dataset metadata.

$$loss = (label - cosine\_sim(u, v))^2 \tag{5.3}$$

Table 17 – Total of matching pairs for each dataset in our experimental setup. We show the values according to the threshold cut-offs (from 0.1 to 0.6). We also point out the number of instances for the whole corpus.

Threshold	NTCIR	ACORDAR
0.1	117k	51k
0.2	69k	37k
0.3	29k	23k
0.4	9k	11k
0.5	2k	4k
0.6	767	1k
Whole Corpus	171k	117k

Source: Created by the author

5.2.2 Retrieval and Ranking

We index the datasets' metadata by applying Elasticsearch.[6] Similar to previous work (BONIFACIO et al., 2022; DAI et al., 2022; JERONYMO et al., 2023), DAPDR retrieves the top-k candidate datasets for each query in the test set using the BM25 similarity function. It then applies a fine-tuned dense retrieval model, built previously in the pipeline, to generate a vector representation for each dataset in the top-k based on its title, description, and keywords, and for the query. DAPDR produces the final ranking for a given text query by sorting the candidate datasets based on the cosine similarity between the query vector and the candidates' vector.

5.3 EXPERIMENTAL SETUP

In this section, we detail our experimental setup for the dataset retrieval task. We describe the datasets (Section 5.3.1), ranking approaches (Section 5.3.2) and the evaluation methodology that we consider (Section 5.3.3).

5.3.1 Datasets

We evaluate DAPDR on NTCIR (KATO et al., 2021) and ACODAR (LIN et al., 2022a), two popular test collections for dataset retrieval. We selected such benchmarks since they have been widely used in other dataset retrieval studies for experimental evaluation. In addition, NTCIR is a famous corpus adopted on the NTCIR-16 Data Search competition, which is a shared task for ad-hoc dataset retrieval.[7] NTCIR contains an English and Japanese corpus crawled from government portals (as e.g., data.gov).[8] We focus on the

[6] https://www.elastic.co
[7] https://ntcir.datasearch.jp
[8] https://www.data.gov

English datasets by using the 2nd edition of the Data Search Task for NTCIR-16,[9] whose corpus contains 46,615 datasets and their metadata and 192 questions. The datasets are available in multiple data formats, including Excel, CSV, XML, JSON and RDF, and the queries are derived from crowd-sourcing workers and question-answering services. ACORDAR includes 31,589 RDF datasets collected from popular open data portals (e.g., *data.medicaid.gov* and *opendata.utah.gov*) and 493 queries. We also consider only its English datasets in our experiments. Similar to the data on NTCIR, human annotators create the queries for ACORDAR, in addition to reusing queries in TREC's Test Collections,[10] since they have great potential for finding relevant data.

5.3.2 Ranking Approaches

Since we target the task of dataset retrieval by using the dataset metadata, which contains text descriptions about its content, we consider IR methods as relevant baselines for our task. In addition, we evaluate a set of general-purpose sentence-embedding models as dense retrieval approaches. Dense retrieval models have shown strong performance for many ranking tasks (CHEN et al., 2020a). We obtain them from Sentence-Transformer API[11], which we summarize as follows:

- **Cos(TD-IDF)** (SALTON; YANG, 1973) It is a basic IR method that represents query and document terms based on term-frequency and inverse document frequency. It ranks the datasets based on cosine distance over the TF-IDF vector. We use TfidfVectorizer from Sklearn for this approach.[12]

- **BM25** (ROBERTSON; ZARAGOZA, 2009) It is an IR algorithm based on the probabilistic relevance framework that uses term-frequency weighting and document length for ranking. We use Rank-BM25 API for the score.[13]

- **DPR** (KARPUKHIN et al., 2020) It is a dense passage retriever based on the BERT architecture. Queries and documents are indexed in a low-dimensional and continuous space using an adapted loss function for positive and negative samples, aiming to produce dense representations such that matching pairs have smaller distances. DPR was originally fine-tuned on Google's Natural Questions dataset.[14]

- **BERT** (DEVLIN et al., 2019) It is a sentence encoder that uses bidirectional transformer networks for language representations, pre-trained on a large corpus. We use a pre-trained BERT from Hugging Face.[15]

[9] https://ntcir.datasearch.jp
[10] https://trec.nist.gov/data/test_coll.html
[11] https://www.sbert.net/docs/pretrained_models.html
[12] https://scikit-learn.org
[13] https://pypi.org/project/rank-bm25
[14] https://ai.google.com/research/NaturalQuestions
[15] https://huggingface.co/docs/transformers/model_doc/bert

- **SBERT** (REIMERS; GUREVYCH, 2019) It is an adapted BERT architecture that uses siamese and triplet networks to derive semantic embeddings for ranking. Using a pooling operation over the BERT output, SBERT produces fixed-sized dense vectors, which can be compared using cosine-similarity. This model was fine-tuned on 1B training pairs using multiple datasets for semantic search.

- **MPNET** (SONG et al., 2020) It is a novel pre-training method that leverages the dependency among predicted tokens through permuted language modeling, in addition to considering the full position information of a sentence during pre-training. Similar to SBERT, MPNET was originally fine-tuned on a 1B sentence pairs dataset.

We train all these models over the augmented data using the fine-tuning strategy described in Section 5.2. The models in Sentence-Transformer API are bi-encoders (DPR, SBERT and MPNET), and we use a cross-encoder approach for BERT. Given a query-dataset pair, the bi-encoders map each side of the input independently to a dense vector, and the cross-encoders perform full attention to the pair (THAKUR et al., 2021).

5.3.3 Retrieval and Ranking Setup

Top-k retrieval. Similar to previous studies (SACHAN et al., 2022), we assume there is an initial pool of datasets for re-ranking. Using BM25 on Elasticsearch, DAPDR retrieves the top-k candidate datasets for each query in the test set. In this setup, we evaluate the cut-offs of $k = 100$ and $k = 1000$. Section 5.4 summarizes the NDCG levels for each retriever using $k = 100$ since it achieves the best retrieval results in our experiments.

Data filtering. We report results with different similarity cut-offs for the data filtering step of DAPDR (0.1, 0.2, 0.3, 0.4, 0.5, and 0.6), in which Section 5.4 outlines the best NDCG levels for each retriever according to their best cut-off setup for NDCG@5.

Evaluation metric. We use NDCG@k at cut-offs $k \in \{5, 10, 15, 20\}$ to measure the quality of the dataset ranking (BONIFACIO et al., 2022). Moreover, we also ran a paired Wilcoxon test to measure the results' significance.

Setup & Implementation. We implement the models using Python 3.6 and TensorFlow 2.2.0, and the experiments are performed on a Titan XP GPU and Ubuntu 16.04 LTS.

5.4 RESULTS AND DISCUSSIONS

In this section, we cover the core results of this chapter. Section 5.4.1 presents the results for dataset retrieval. An ablation study on the dataset metadata is detailed in Section 5.4.2, and Section 5.4.3 explores the filtering strategy of our pipeline. Finally, in Section 5.4.4, we summarize the type of queries created by the LLMs.

5.4.1 Results for Dataset Retrieval

Table 18 depicts the NDCG@k for NTCIR and ACORDAR benchmarks. The numbers show that the fine-tuned models surpass the other original models for both corpora. The fine-tuned MPNET (F_MPNET) obtains the best NDCG@k for all cut-offs on NTCIR, and the fine-tuned BERT (F_BERT) attains the best values on ACORDAR.

Furthermore, comparing the original and fine-tuned strategies, the fine-tuned DPR, SBERT, MPNET, and BERT outperform their original model for both benchmarks in all NDCG@k values. For example, on the NTCIR corpus, the fine-tuned DPR, SBERT, and MPNET enhance by over 69%, 21%, and 6% of NDCG@5 respectively, when compared to the original approach. The fined-tuned BERT achieves the highest margin, being three times better compared to its original one. The results go similarly on the ACORDAR dataset, in which the fine-tuned DPR, SBERT, MPNET and BERT improve by over 37%, 10%, 9% and 325% respectively for NDCG@5 in contrast to their pre-trained models. It is worth noting that even highly tuned models for similarity, such as SBERT and MPNET, originally trained on 1B similarity pairs, improve their performance by fine-tuning on DAPDR's generated instances.

To validate the results, we perform a Paired-Wilcoxon Test for NDCG@5. Specifically, we compare each fine-tuned model against its original approaches, in which our alternative hypothesis is the fine-tuned strategies have greater NDCG@5 values than the original ones. Table 19 shows the p-values for each comparison in both benchmarks. The overall results show that most fined-tuned models statistically outperform their original approaches regarding NDCG@5 (i.e., bold values for p-values < 0.05). For example, F_DPR, F_SBERT and F_BERT statistically surpass their original models on both NTCIR and ACORDAR datasets. The same is true for F_MPNET in the ACORDAR

Table 18 – Our core results for NTCIR and ACORDAR benchmarks. We point out the NDCG@k for each original model (O) and fine-tuned (F) approach and the IR baselines. Bold values stand for the best results of each metric.

Approach	**NTCIR Dataset - *Recall@100***				**ACORDAR Dataset - *Recall@100***			
	NDCG@5	**10**	**15**	**20**	**NDCG@5**	**10**	**15**	**20**
BM25	0.1725	0.1739	0.1757	0.1861	0.4026	0.4094	0.4209	0.4354
Cos(TF-IDF)	0.2105	0.2058	0.2129	0.2255	0.4206	0.4269	0.4391	0.4531
O_DPR	0.1423	0.1435	0.1543	0.1643	0.2723	0.2885	0.3089	0.3246
O_SBERT	0.1991	0.1995	0.2091	0.2221	0.3548	0.3598	0.3730	0.3908
O_MPNET	0.2659	0.2581	0.2686	0.2825	0.3875	0.3958	0.4133	0.4317
O_BERT	0.0620	0.0688	0.0735	0.0832	0.1048	0.1153	0.1308	0.1449
F_DPR	0.2415	0.2460	0.2530	0.2661	0.3736	0.3806	0.3972	0.4182
F_SBERT	0.2411	0.2338	0.2396	0.2507	0.3935	0.3996	0.4136	0.4302
F_MPNET	**0.2827**	**0.2763**	**0.2784**	**0.2897**	0.4239	0.4265	0.4434	0.4600
F_BERT	0.2561	0.2558	0.2664	0.2779	**0.4460**	**0.4461**	**0.4590**	**0.4745**

Source: Created by the author

Table 19 – Paired Wilcoxon Test for NDCG@5. We compare each fine-tuned model (F) against the original ones (O). Our alternative hypothesis is that they have greater performance them the original models ($F > O$). Bold values stand for those comparisons in which we can reject the null hypothesis, p-value < 0.05.

NTCIR Dataset				
Vs.	O_DPR	O_SBERT	O_MPNET	O_BERT
F_DPR	**8.38e-07**	**0.002717**	0.993851	**2.91e-16**
F_SBERT	**1.73e-06**	**0.002192**	0.958568	**6.09e-15**
F_MPNET	**5.32e-12**	**8.09e-08**	0.069817	**1.33e-18**
F_BERT	**2.83e-09**	**1.26e-05**	0.795181	**1.33e-18**

ACORDAR Dataset				
Vs.	O_DPR	O_SBERT	O_MPNET	O_BERT
F_DPR	**4.19e-13**	0.153745	0.842155	**1.97e-39**
F_SBERT	**1.46e-16**	**1.73e-05**	0.332389	**4.94e-44**
F_MPNET	**4.28e-25**	**6.30e-08**	**1.87e-08**	**2.13e-51**
F_BERT	**1.46e-29**	**6.41e-12**	**8.57e-06**	**1.46e-52**

Source: Created by the author

benchmark. The only exception is F_MPNET on the NTCIR dataset, in which we can not reject the null hypothesis since their p-value is 0.069817. In summary, the results confirm that most dense retrieval models improve their NDCG@5 performance using DAPDR.

Comparing the best model for each benchmark against BM25 and Cos(TF-IDF), F_MPNET statistically outperforms both BM25 and Cos(TF-IDF) on NTCIR, whose p-values are 2.50e-07 and 1.64e-05, respectively. The F_BERT goes analogous on ACORDAR, whose p-values are 5.66e-05 for BM25 and 0.00850 for Cos(TF-IDF).

5.4.2 Ablation Study on the Dataset Metadata

Since we target dataset retrieval by using the dataset metadata, which typically includes title, description, and keywords, we first evaluate distinct combinations in the context of ranking. Table 20 shows the NDCG@5 values for each setup on both evaluation datasets and retrieval models. Except for SBERT in NTCIR dataset, whose combination of *<title>* and *<tags>* achieves the highest results, the concatenation of the dataset *<title>*, *<description>* and *<keywords>* obtains the best NDCG@5 values for all other models. The *<tags>* achieves the worst for the same metric. Lastly, it is worth mentioning that the combination of *<title>* and *<description>* attains relevant results for this task when compared to the other ones. On average, the NDCG@5 results improve by up 1% to 4% when we combine all the metadata as retrieval features, in contrast to combining two of them (i.e., *<title>* and *<description>*).

Table 20 – Ablation study for the dataset metadata. We evaluate distinct combinations for the fine-tuning strategy. The NDCG@5 is presented for each retrieval approach and metadata setup. Bold values point out the best results for each model. The concatenation of the metadata title, description and keywords attains the best results for most retrieval models.

Dataset Metadata	NTCIR Dataset *NDCG@5*				ACORDAR Dataset *NDCG@5*			
	DPR	**SBERT**	**MPNET**	**BERT**	**DPR**	**SBERT**	**MPNET**	**BERT**
Title	0.2342	0.2308	0.2572	0.2141	0.3487	0.3659	0.3869	0.3864
Description	0.2188	0.1947	0.2573	0.2410	0.3260	0.3306	0.3851	0.3832
Tags	0.2007	0.1802	0.2144	0.1819	0.2735	0.2728	0.2923	0.2783
Title, Descritpion	0.2310	0.2235	0.2744	0.2467	0.3657	0.3920	0.4152	0.4402
Title, Tags	0.2320	**0.2411**	0.2679	0.2289	0.3612	0.3736	0.3900	0.4113
Description, Tags	0.2191	0.2104	0.2550	0.2506	0.3355	0.3452	0.4004	0.3996
Title, Description, Tags	**0.2415**	0.2264	**0.2827**	**0.2561**	**0.3736**	**0.3935**	**0.4239**	**0.4460**

Source: Created by the author

5.4.3 Ablation Study on the Data Filtering Step

Not limited to the dataset metadata evaluation, we also analyze the data filtering step from our pipeline. Specifically, we also fine-tune each retrieval model by applying a distinct training corpus according to each similarity cut-offs (0.1, 0.2, 0.3, 0.4, 0.5, and 0.6). At this front, our goal is to evaluate the retrievers tuned on the filtered data and compare their performance against the models tuned into the whole corpus. Table 21 shows the NDCG@5 results for each training data and for each retrieval approach in both benchmarks.

In summary, the ranking results of this task are improved when we fine-tune the models on the filtered data as, e.g., for the training data using the similarity thresholds 0.1, 0.2, 0.3 and 0.4 in both NTCIR and ACORDAR datasets. In fact, the retrievers attain their best NDCG@5 levels by using this training corpus. In contrast, the NDCG@5 decreases for several of them when we apply the highest cut-off (i.e., 0.6).

Finally, such results also confirm the efficacy of our filtering step since the best values for NDCG@5 are obtained by tuning the retrievers on the clean data for low similarity cut-offs. On average, compared to the whole corpus, the models tuned into our filtered data improved their results from 6% to 17% in terms of NDCG@5.

5.4.4 Analysis of the Synthetic Queries

Similar to previous work (KACPRZAK et al., 2017; KACPRZAK et al., 2018; KACPRZAK et al., 2019), we also investigate the most common types of queries produced by *docTTTTTquery* in our data augmentation task. Table 22 summarizes the rate of queries by each type/format in the utilized benchmarks.

For both datasets, direct questions are the most frequent, i.e., queries for the *<what>* format as, e.g., *what causes oregon murres die*, which focuses on data about classes of deaths for seabirds. *<location>* questions are also present in both corpora, over 13% for NTCIR and 16% for ACORDAR. The study of Kacprzak et al. (2019) shows that intent-

Table 21 – Ablation study on the training corpus. We evaluate different similarity cut-offs for the data filtering step in our pipeline. We present the NDCG@5 for each training data as well as for each retrieval model. Bold values mean the best results for each approach. Note that we also include the "*Whole Corpus*" training data for comparison purposes.

Training Corpus	**NTCIR Dataset** *NDCG@5*				**ACORDAR Dataset** *NDCG@5*			
	DPR	**SBERT**	**MPNET**	**BERT**	**DPR**	**SBERT**	**MPNET**	**BERT**
Whole Corpus	0.2061	0.2179	0.2461	0.2410	0.3493	0.3677	0.3775	0.4163
Similarity > 0.1	0.2132	**0.2411**	0.2392	0.2531	0.3611	0.3794	0.3864	0.4252
Similarity > 0.2	0.2329	0.2407	0.2522	**0.2561**	0.3606	0.3873	0.3987	0.4208
Similarity > 0.3	**0.2415**	0.2275	0.2678	0.2398	**0.3736**	0.3906	0.3973	0.4387
Similarity > 0.4	0.2141	0.2217	**0.2827**	0.2247	0.3729	**0.3935**	0.4179	**0.4460**
Similarity > 0.5	0.1648	0.2151	0.2635	0.2058	0.3287	0.3704	**0.4239**	0.3965
Similarity > 0.6	0.1439	0.2048	0.2654	0.2107	0.2777	0.3546	0.3959	0.3917

Source: Created by the author

Table 22 – Analysis of the synthetic queries. We evaluate the types of queries produced by *docTTTTTquery*. We summarize their most common formats including what, where, location and so on. We sort the rates by NTCIR corpus.

Queries Types	**NTCIR (%)**	**ACORDAR (%)**
What	42.47	30.37
Location	13.39	16.01
When	10.54	6.23
Which	2.67	1.77
Who	1.71	1.90
How	1.68	3.07
Why	1.38	0.24
Numerical	1.04	3.70
How Many	0.36	1.49

Source: Created by the author

queries for geospatial attributes are quite common on open-data portals. They contain information about countries, cities, and regions. We use the Location Tagger API to detect locations in the queries.[16], or queries that include the token *<where>* as, e.g., *where is trans alaska gas terminals*, which covers data about oil transportation systems in Alaska.

In addition, questions using *<when>* are most predominant in the NTCIR dataset. One example is the query: *when was nos hydrographic survey conducted*, which contains data for the NOS survey. Lastly, the other kinds of queries, such as <which>, <who>, or <how many>, have the lowest rates in the corpus (over 1% to 3%). The same is true for the <numerical> questions, which can contain time queries such as, e.g., *IEC election 2014*, which covers data for election results. Such analysis also confirms that the model *docTTTTTquery* can produce diverse types of queries for data augmentation in the context of dataset retrieval task.

[16] https://pypi.org/project/locationtagger

5.5 CONCLUDING REMARKS

In this chapter, we employed LLMs for query augmentation in the dataset search task since queries mapped to datasets are very limited in most evaluation benchmarks. Dataset search is an information retrieval task that aims to find relevant datasets to a user query. Specifically, we introduced DAPDR (Data Augmentation Pipeline for Dataset Retrieval): a solution that uses LLMs to create training samples for dense retrievers. The core of our approach was to use the augmented query-dataset pairs for fine-tuning the dense retrievers for ranking. DAPDR was also assessed using a set of sentence-embedding approaches for semantic search over NTCIR and ACORDAR benchmarks, and our goal was to evaluate whether the models tuned into DAPDR outperform the original models. In our experiments, we evaluated the retrievers DPR, BERT, SBERT and MPNET, which have been utilized for many ranking tasks.

The overall results pointed out that the retrievers tuned on DAPDR statistically outperform the original models by a large margin, also surpassing the IR baselines BM25 and Cos(TF-IDF). On average, our results for this task are improved from 6% to 69% in terms of NDCG@5 by tuning the retrievers on our pipeline. Furthermore, in order to validate our approach, we also presented an ablation study on the dataset metadata and on the data filtering step of DAPDR, in addition to evaluating the type of queries created by the *docTTTTTquery*. The models tuned into the filtered corpus attained the best results for NDCG@5 compared to those trained in the whole data. Regarding the dataset metadata, the concatenation of *<title>*, *<description>* and *<keyword>* achieved the best results for this task in contrast to the other metadata setups. Finally, we also confirmed that *docTTTTTquery* produced distinct types of queries for our task, including searches for geospatial queries. Such evaluations also proved that LLMs can generate relevant training samples in the context of dataset retrieval.

6 CONCLUSION

In this thesis work, we have addressed IR problems for structured/semi-structured data retrieval. By considering the domain of Web Tables, which comprises a huge and rich corpus of relational data from the Internet, we focused on the following tasks: QA Table Retrieval, News-Table Matching, and Dataset Retrieval. At this direction, we also discussed that one of the main challenges for such tasks is that they need to match unstructured queries and structured/semi-structured tables, since each task utilizes relational data to represent information. Furthermore, another point examined in this work was the development of adapted matching models, whose goal was to compute the similarity degree between queries and tables. Given this gap, we also formalized these problems as an adjusted ranking task, where we aimed to find a set of relevant tables or datasets for user queries and news stories.

Towards this front, in Chapter 3 of this thesis, we have introduced a novel taxonomy for QA table retrieval, which surveyed 16 core studies for this task. As a result, we classified the solutions for table retrieval into five groups: probabilistic-based, IR-based, entity-based, feature-based, and network-based. Our work also pointed out a precise ablation study on popular table features for retrieval approaches, which was organized into query-dependent and query-independent attributes, document fields and document measures. Moreover, this chapter also presented the anatomy of the tables on the Web, in addition to suggesting a set of open challenges for this task and popular benchmarks for experimental evaluations. Lastly, in order to fill the research questions concerning this query-table match, we make the following conclusions. Regarding RQ1 (*How can we retrieve contextual tables for a query from a table corpus?*), we argue that the designed methods for this task apply a table retrieval solution according to two cascade steps: (I) they first efficiently find a pool of candidate tables by applying traditional IR algorithms such as BM25; and (II) they re-rank this table subset by adopting sophisticated approaches build on top of neural networks. For RQ2 (*How to calculate the similarity between unstructured queries and semi-structured/structured Web Tables?*), we claim that the relevant alternatives for this task extract a set of query-table features based on neural network components (such as Long Short-Term Memory (LSTM) layers, Gated Recurrent Units (GRUs) and Transformer-Based Architectures). Concerning RQ3 (*Which table aspects such as headers, caption or body are more relevant to the match in the context of table retrieval?*), we show that popular features for this task include table caption, headers and body, in addition to the surrounding table features (as e.g., page title and reference text).

In sequence, Chapter 4 addressed the news-table matching problem. This task was first introduced as a novel table retrieval application, whose Web Tables are employed to expand the content of a news story for news augmentation. Towards this goal, our

core contribution was a BERT-based model that combines recurrent networks, attention mechanisms and transformer layers in a single neural network. In addition, we have also assessed the efficacy of traditional IR methods, document/sentence encoders, neural IR models and dense retrievers in the context of this task. In resume, a hypothesis test has demonstrated that our solution statistically outperforms all the other baselines in terms of the mean reciprocal rank metric. Besides, in this chapter, we also presented the first news-table corpus from literature. By crawling Wikipedia pages, we collected 275,352 news articles and 298,792 web tables, whose ground truth contains 93,818 matching pairs created by distant supervision strategies. Lastly, regarding the research question RQ4 (*How to compute the matching degree between News Stories and Web Tables for the news-table matching task?*), we argue that the news-table similarity can be calculated by matching the contextual information of the tables against the news title or short description.

Finally, in Chapter 5 of this thesis, we concentrated on the dataset retrieval task. Similar to table retrieval, the goal of this task was to rank structure datasets for a text query, in which a dataset is defined as a collection of relational data classified for a particular intent. Besides, queries for this task are generally formed by geographical places or temporal data. Specifically for this domain, our goal was to utilize LLMs to generate supervised training data for fine-tuning dense retrievers, as labeled queries for this task are very scarce in most benchmarks, such as ACORDAR and NTCIR. For that, we introduced DAPDR (Dataset Retrieval Pipeline), which generates synthetic queries of dataset descriptions, later utilized in a fine-tuning task. DAPDR was evaluated by using a set of dense retrievers for semantic search over popular dataset retrieval benchmarks, and we aimed to evaluate whether the models tuned into DAPDR outperform the other baselines. In summary, we confirmed that the retrievers tuned in DAPDR have surpassed the original models at different NDCG levels.

Based on that, in order to fill the research question RQ5 (*Can we improve the efficacy of supervised retrievers by fine-tuning them on LLMs augmented queries for the dataset retrieval task?*), we claim that the models tuned into LLM data, specifically for this task, have better efficacy in contrast to the original approaches. In addition, we also point out that even highly tuned models for similarity tasks, such as SBERT and MPNET, which were originally trained on a huge corpus of similarity pairs, improve their ranking performance by fine-tuning on DAPDR's generated queries. Lastly, we also argue that no retriever reduces its ranking accuracy compared to the original model when tuned on the LLM synthetic data. Towards this front, popular open-data portals, such as *dados.gov.br* from Brazil,[1] can benefit from our dataset retrieval pipeline by applying dense retrievers to their search methodology, also using LLMs for producing synthetic queries for training purposes. Besides, using DAPDR, such platforms could expand their methodology from lexical matching to semantic matching to obtain the best datasets for a user query.

[1] https://dados.gov.br

As follows, we outline this thesis contributions and publications, also presenting its limitations and future work.

6.1 SUMMARY OF CONTRIBUTIONS

We summarize the contributions of this thesis as follows, which we categorize according to each task domain: QA Table Retrieval, News-Table Matching and Dataset Retrieval.

- **QA Table Retrieval**
  - We present a novel taxonomy for the task of QA table retrieval that classifies the proposed methods into five groups, from probabilistic approaches to novel sophisticated deep learning architectures: probabilistic-based, IR-based, entity-based, feature-based, and network-based;
  - We present a detailed ablation study on popular table features for retrieval, which we classify into query-dependent and query-independent aspects, as well as document fields and document measures.
- **News-Table Matching**
  - We introduce the problem of collocating news articles with structured Web Tables as a novel ranking task. Furthermore, we also formalize the most used matching features for news-table matching;
  - We present the first news-table corpus from literature. By crawling Wikipedia pages, we collected 275,352 news articles and 298,792 web tables, whose ground truth contains 93,818 matching pairs;
  - We evaluate previous approaches for table retrieval and table matching in the context of this task, also assessing both single and multi-field (document) ranking methodologies in the experiments;
  - We propose a novel BERT-based attention model for computing the similarity degree between news stories and structured tables;
  - We compare the performance of our solution with IR techniques, sentence encoders, text matching models and neural approaches for this task;
- **Dataset Retrieval**
  - We introduce DAPDR, a dataset retrieval pipeline that uses LLMs to generate labeled queries for dataset descriptions, which are then utilized for fine-tuning dense retrievers at the target task.
  - We evaluate DAPDR on a set of popular retrievers for semantic search over dataset search benchmarks, in addition to assessing the performance of traditional IR methods in the context of this task.

6.2 NTCIR-16 CHALLENGE

Another contribution of this thesis work is our participation in a dataset retrieval challenge, which was organized by NTCIR Conference (NTCIR-16).[2] This edition included a shared task on ad-hoc dataset retrieval for governmental data, which contained collections gathered from data published by the US government (data.gov) and the Japanese government (e-Stat). Our team (NYUCIN) was a joint effort of members from the Centro de Informática (CIn) at Universidade Federal de Pernambuco (UFPE), and the Tandon School of Engineering at New York University (NYU). We addressed the dataset retrieval task for English datasets.

In summary, the official results of the competition showed that our approach achieved the highest score among all submitted runs for all evaluation metrics (KATO et al., 2022). NTCIR-16 has received 37 submissions for the English datasets, separated into 6 groups. In particular, our results represent a 30% improvement compared to the second-best one in the competition. For our submission, we used the proposed news-table matching model (introduced in Chapter 4), adapted to the dataset retrieval task (SILVA et al., 2022).

6.3 LIST OF PUBLICATIONS

This section summarizes the publications of this thesis work. We have been submitting the core results of this thesis to international/national journals and conferences in the computer science field. As follows, we list accepted papers as well as those studies that are still under review.

1. Accepted Papers:

    - **SILVA, LEVY**; BARBOSA, LUCIANO. **Matching News Articles and Wikipedia Tables for News Augmentation**. Knowledge and Information Systems, v. 65, p. 1713-1734, 2023.
    - LEES, A.; BARBOSA, L.; KORN, F.; **SILVA, L**. S.; WU, Y.; YU, C. **Collocating News Articles with Structured Web Tables**. In: 1st International Workshop on News Recommendation and Intelligence, 2021, Ljubljana, Slovenia. Companion Proceedings of the Web Conference 2021. New York: ACM, 2021. p. 393-401.
    - **SILVA, L.**; BARBOSA, L.; CASTELO, S.; ZHANG, H.; SANTOS, A.; FREIRE, J. **NYUCIN at the NTCIR-16 Dataset Search 2 Task**. In: The 16th NTCIR Conference Evaluation of Information Access Technologies, 2022, Tokyo, Japan. Proceedings of the 16th NTCIR Conference on Evaluation of Information Access Technologies, 2022. p. 38-45.

[2] https://ntcir.datasearch.jp/subtasks/ir

2. Under Review:

    - **SILVA, LEVY**; BARBOSA, LUCIANO. **A Survey on Intelligent Solutions for Table Retrieval**. Submitted to Journal of the Brazilian Computer Society.
    - **SILVA, LEVY**; BARBOSA, LUCIANO. **Improving Dense Retrieval Models with LLM Augmented Data for Dataset Search**. Submitted to Journal of Knowledge-Based Systems.

6.4 LIMITATIONS

In this section, we point out some limitations of this study, as follows.

- **Retrieval Corpus.** In this thesis work, we have utilized the Elasticsearch framework to retrieve an initial pool of the $top - 100$ candidate tables or datasets for re-ranking. However, in our experimental setup, this subset does not contain all matching tables/datasets according to our labeled data (i.e., the maximum recall scenario), which limits the results for our re-ranking models and baselines.

- **Prediction Time**. According to our experimental setup, our BERT-based attention model has the most elevated prediction time compared to the other methodologies for the news-table matching task. In fact, cross-encoder models, similar to our solution, have elevated inference time since they need to perform full attention to the input pair. One alternative for that is to use a distilled version of BERT instead of the original one, which has 40% fewer parameters than the BERT model.

- **Wikipedia Matching Tables**. In this study, we fine-tuned our BERT-based solution as well as the other baselines by using a set of matching tables created by a distant supervision strategy, since we do not have any labeled corpus for the news-table matching task. Based on that, one limitation of this approach is that our matching tables are chosen according to a similarity threshold, and our manual analysis showed that about 8% of this data is a non-matching sample. As a result, the training dataset can also contain false positive examples.

- **Metadata Information**. For both of our solutions, the DAPDR pipeline as well as the BERT-based attention model, we only include the surrounding text of the structured corpus as the matching features for retrieval as, e.g., the title, short description and keywords for tables or datasets. As a result that, the table body (i.e., its lines and rows) are not incorporated into our retrieval approaches, which restricts our solutions for retrieving data-driven intent queries for this task.

- **DocTTTTTquery Model**. One limitation of *docTTTTTquery* is that it mostly produces questions for the "what" format, which limits the training of the supervised retrievers for other types of queries as, e.g., for temporal or numerical questions in the context of dataset retrieval. Such types of queries are also present in the dataset retrieval domain, but *docTTTTTquery* creates a limited sample of them.

- **DAPDR Filtering Step**. One restriction of the DAPDR pipeline is that we need to manually select a similarity threshold for pruning the query-dataset pairs for producing the final fine-tuning corpus. However, our experimental setup has demonstrated that various similarity cut-offs have achieved relevant results for distinct retrievers and benchmarks (NTCIR and ACORDAR). As a result, more ablation studies on this step of our pipeline are necessary to improve the DAPDR accuracy.

- **Dataset Search**. In this thesis work, we have only focused on a query augmentation topic for the dataset retrieval task, since labeled data is very scarce for this task. Based on that, we do not explore other core challenges for this task including: (I) a deep analysis of the dataset queries issued on open data portals; (II) the development of adapted matching models for this task; and (III) a profound investigation of the types and formats in which datasets are published on the web.

6.5 FUTURE WORK

In this section, we suggest future studies for complementing this thesis work, as follows.

- **Retrieval Approach**. For both table/dataset retrieval approaches we proposed in this thesis, we employed the BM25 algorithm as the default Elasticsearch method. Based on that, in order to enhance the set of retrieval candidates (usually the top-100 matching tables or datasets), further retrieval solutions can also utilize dense retrievers at this step of our pipeline.

- **Table Body**. In our experimental setup, we have used the surrounding text of tables/datasets as the retrieval attributes (for both BERT-based model and DAPDR pipeline), i.e., the corpus metadata such as its title, description, or keywords. Based on that, a point of improvement towards this goal is to include the table body in the ranking approaches, by extracting and including its lines and rows in the solutions.

- **Semantic Features**. In order to enhance the performance of the BERT-based approach, future work can also add more semantic features to the network architecture, in addition to the news title, description and keyword, such as entities or categories.

- **Mismatch Articles**. In this work, we only focused on matching tables for news articles. However, we do not estimate the number of news articles that may not be able to match web tables. Future studies are necessary for this perspective.

- **News Improvement**. In this thesis work, we have argued that structured Web Tables can enhance the content of a news story. However, in our experimental evaluation, we do not explore the improvement of the news understanding brought by the top-ranked Web Tables. Further analysis is also necessary on this front.

- **Query Generation**. The model *docTTTTTquery* is a zero-shot query generation approach for text data augmentation. However, in our experimental analysis, this model mostly produced direct questions for the dataset descriptions. Given this gap, a point of advancement is to explore prompting strategies in the LLM for creating specific types of queries as, e.g., for geographical or temporal queries.

- **Large Language Models**. In order to compare *docTTTTTquery* against different LLMs, future work can explore other LLMs in the query generation step of DAPDR, such as Falcon[3] or OpenLLaMA[4] models, which are also open-source approaches. In addition, since LLMs can increase task performance when using large amounts of data, future studies could also explore the LLM's performance characteristics, since latency is problematic for real-time applications.

- **Dataset Snapshots**. In DAPDR pipeline, the statistical data from the dataset is not used in their ranking approach, which limits the search for some types of queries. Based on that, other studies can also apply LLMs for generating text snapshots from the dataset body to also cover data-driven questions.

- **BERT-Based Model**. In the context of the dataset retrieval task, we do not develop adapted matching models for queries and datasets. Based on that, future work can also explore this gap. An alternative is to fine-tune our BERT-based attention model by using the augmented LLM data for the dataset retrieval task.

[3] https://huggingface.co/tiiuae/falcon-7b
[4] https://github.com/openlm-research/open_llama

REFERENCES

AGARWAL, S.; SINGH, N. K.; MEEL, P. Single-document summarization using sentence embeddings and k-means clustering. In: IEEE. *Proceedings of the 2018 International Conference on Advances in Computing, Communication Control and Networking*. Greater Noida, India, 2018. p. 162–165.

AKASH, A. U.; NAYEEM, M. T.; SHOHAN, F. T.; ISLAM, T. Shironaam: Bengali news headline generation using auxiliary information. In: *Proceedings of the 17th Conference of the European Chapter of the Association for Computational Linguistics*. Dubrovnik, Croatia: Association for Computational Linguistics, 2023. p. 52–67.

ALLAM, A. M. N.; HAGGAG, M. H. The question answering systems: A survey. *International Journal of Research and Reviews in Information Sciences*, Science Academy, v. 2, n. 3, 2012.

ATRI, Y. K.; GOYAL, V.; CHAKRABORTY, T. Multi-document summarization using selective attention span and reinforcement learning. *IEEE ACM Transactions on Audio, Speech, and Language Processing*, v. 31, p. 3457–3467, 2023.

BALAKRISHNAN, S.; HALEVY, A. Y.; HARB, B.; LEE, H.; MADHAVAN, J.; ROSTAMIZADEH, A.; SHEN, W.; WILDER, K.; WU, F.; YU, C. Applying webtables in practice. In: *Proceedings of the 7th Biennial Conference on Innovative Data Systems Research*. Asilomar, California: CIDR, 2015.

BENJELLOUN, O.; CHEN, S.; NOY, N. F. Google dataset search by the numbers. In: PAN, J. Z.; TAMMA, V. A. M.; D'AMATO, C.; JANOWICZ, K.; FU, B.; POLLERES, A.; SENEVIRATNE, O.; KAGAL, L. (Ed.). *Proceedings of the 19th International Semantic Web Conference*. Athens, Greece: Springer, 2020. (Lecture Notes in Computer Science, v. 12507), p. 667–682.

BERGER, A. L.; LAFFERTY, J. D. Information retrieval as statistical translation. In: *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Berkeley, California: ACM, 1999. p. 222–229.

BHAGAVATULA, C. S.; NORASET, T.; DOWNEY, D. Methods for exploring and mining tables on wikipedia. In: *Proceedings of the ACM SIGKDD Workshop on Interactive Data Exploration and Analytics*. Chicago, Illinois: ACM, 2013. p. 18–26.

BHAGAVATULA, C. S.; NORASET, T.; DOWNEY, D. Tabel: Entity linking in web tables. In: *Proceedings of the 14th International Semantic Web Conference*. Bethlehem, Pennsylvania: Springer, 2015. (Lecture Notes in Computer Science, v. 9366), p. 425–441.

BHATTACHARYYA, A. On a measure of divergence between two multinomial populations. *Sankhyā: The Indian Journal of Statistics*, Journal Storage, p. 401–406, 1946.

BONIFACIO, L. H.; ABONIZIO, H.; FADAEE, M.; NOGUEIRA, R. F. Inpars: Data augmentation for information retrieval using large language models. *CoRR*, abs/2202.05144, 2022. Available at: <https://arxiv.org/abs/2202.05144>.

BROWN, T. B.; MANN, B.; RYDER, N.; SUBBIAH, M.; KAPLAN, J.; DHARIWAL, P.; NEELAKANTAN, A.; SHYAM, P.; SASTRY, G.; ASKELL, A.; AGARWAL, S.; HERBERT-VOSS, A.; KRUEGER, G.; HENIGHAN, T.; CHILD, R.; RAMESH, A.; ZIEGLER, D. M.; WU, J.; WINTER, C.; HESSE, C.; CHEN, M.; SIGLER, E.; LITWIN, M.; GRAY, S.; CHESS, B.; CLARK, J.; BERNER, C.; MCCANDLISH, S.; RADFORD, A.; SUTSKEVER, I.; AMODEI, D. Language models are few-shot learners. In: LAROCHELLE, H.; RANZATO, M.; HADSELL, R.; BALCAN, M.; LIN, H. (Ed.). *Proceedings of the 33rd Advances in Neural Information Processing Systems*. Virtual Event: Curran Associates, 2020. v. 33, p. 1877–1901.

CAFARELLA, M. J.; HALEVY, A. Y.; LEE, H.; MADHAVAN, J.; YU, C.; WANG, D. Z.; WU, E. Ten years of webtables. *Proceedings of the VLDB Endowment*, v. 11, n. 12, p. 2140–2149, 2018.

CAFARELLA, M. J.; HALEVY, A. Y.; WANG, D. Z.; WU, E.; ZHANG, Y. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment*, v. 1, n. 1, p. 538–549, 2008.

CAFARELLA, M. J.; HALEVY, A. Y.; WANG, D. Z.; WU, E.; ZHANG, Y. Webtables: exploring the power of tables on the web. *Proceedings of the VLDB Endowment*, v. 1, n. 1, p. 538–549, 2008.

CAFARELLA, M. J.; HALEVY, A. Y.; ZHANG, Y.; WANG, D. Z.; WU, E. Uncovering the relational web. In: *Proceedings of the 11th International Workshop on the Web and Databases*. Vancouver, Canada: IEEE, 2008. p. 1–6.

CAI, P.; SONG, K.; CHO, S.; WANG, H.; WANG, X.; YU, H.; LIU, F.; YU, D. Generating user-engaging news headlines. In: *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics*. Toronto, Canada: Association for Computational Linguistics, 2023. v. 1, p. 3265–3280.

CALLAN, J. P.; CROFT, W. B.; HARDING, S. M. The INQUERY retrieval system. In: *Proceedings of the International Conference on Database and Expert Systems Applications*. Valencia, Spain: Springer, 1992. p. 78–83.

CARMEL, D.; CHANG, M.; GABRILOVICH, E.; HSU, B. P.; WANG, K. Erd'14: entity recognition and disambiguation challenge. In: *Proceedings of the 37th International ACM SIGIR Conference on Research and Development in Information Retrieva*. Gold Coast, Australia: ACM, 2014. p. 1292.

CASTELO, S.; RAMPIN, R.; SANTOS, A. S. R.; BESSA, A.; CHIRIGATI, F.; FREIRE, J. Auctus: A dataset search engine for data discovery and augmentation. *Proceedings of the VLDB Endowment*, v. 14, n. 12, p. 2791–2794, 2021.

CER, D.; YANG, Y.; KONG, S.-y.; HUA, N.; LIMTIACO, N.; JOHN, R. S.; CONSTANT, N.; GUAJARDO-CÉSPEDES, M.; YUAN, S.; TAR, C. et al. Universal sentence encoder. *CoRR*, abs/1803.11175, 2018. Available at: <http://arxiv.org/abs/1803.11175>.

CHAKRABARTI, K.; CHEN, Z.; SHAKERI, S.; CAO, G.; CHAUDHURI, S. Tableqna: Answering list intent queries with web tables. *CoRR*, abs/2001.04828, 2020. Available at: <https://arxiv.org/abs/2001.04828>.

CHAPMAN, A.; SIMPERL, E.; KOESTEN, L.; KONSTANTINIDIS, G.; IBÁÑEZ, L.; KACPRZAK, E.; GROTH, P. Dataset search: a survey. *Proceedings of the VLDB Endowment*, v. 29, n. 1, p. 251–272, 2020.

CHEN, Z.; JIA, H.; HEFLIN, J.; DAVISON, B. D. Leveraging schema labels to enhance dataset search. In: JOSE, J. M.; YILMAZ, E.; MAGALHÃES, J.; CASTELLS, P.; FERRO, N.; SILVA, M. J.; MARTINS, F. (Ed.). *Proceedings of the 42nd European Conference on Information Retrieval*. Lisbon, Portugal: Springer, 2020. (Lecture Notes in Computer Science, v. 12035), p. 267–280.

CHEN, Z.; TRABELSI, M.; HEFLIN, J.; XU, Y.; DAVISON, B. D. Table search using a deep contextualized language model. In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Virtual Event: ACM, 2020. p. 589–598.

CHO, K.; MERRIENBOER, B. van; GÜLÇEHRE, Ç.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Doha, Qatar: ACL, 2014. p. 1724–1734.

CHO, K.; MERRIENBOER, B. van; GÜLÇEHRE, Ç.; BAHDANAU, D.; BOUGARES, F.; SCHWENK, H.; BENGIO, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. In: *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing*. Doha, Qatar: ACL, 2014. p. 1724–1734.

CHUNG, J.; GÜLÇEHRE, Ç.; CHO, K.; BENGIO, Y. Empirical evaluation of gated recurrent neural networks on sequence modeling. *CoRR*, abs/1412.3555, 2014. Available at: <http://arxiv.org/abs/1412.3555>.

CHURCH, K. W.; HANKS, P. Word association norms, mutual information and lexicography. In: *Proceedings of the 27th Annual Meeting of the Association for Computational Linguistics*. Vancouver, Canada: ACL, 1989. p. 76–83.

CRASWELL, N. *Encyclopedia of Database Systems*. New York: Springer, 2018.

DAI, Z.; XIONG, C.; CALLAN, J.; LIU, Z. Convolutional neural networks for soft-matching n-grams in ad-hoc search. In: *Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. Marina Del Rey, California: ACM, 2018. p. 126–134.

DAI, Z.; ZHAO, V. Y.; MA, J.; LUAN, Y.; NI, J.; LU, J.; BAKALOV, A.; GUU, K.; HALL, K. B.; CHANG, M. Promptagator: Few-shot dense retrieval from 8 examples. *CoRR*, abs/2209.11755, 2022. Available at: <https://doi.org/10.48550/arXiv.2209.11755>.

DENG, L.; ZHANG, S.; BALOG, K. Table2vec: Neural word and entity embeddings for table population and retrieval. In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Paris, France: ACM, 2019. p. 1029–1032.

DEVLIN, J.; CHANG, M.-W.; LEE, K.; TOUTANOVA, K. 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*. Minneapolis, Minnesota: Association for Computational Linguistics, 2019. p. 4171–4186.

DIAZ, F.; MITRA, B.; CRASWELL, N. Query expansion with locally-trained word embeddings. In: *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. Berlin, Germany: The Association for Computer Linguistics, 2016. p. 367–377.

EGGERT, G.; HUO, K.; BIVEN, M.; WAUGH, J. Tablib: A dataset of 627m tables with context. *CoRR*, abs/2310.07875, 2023. Available at: <https://doi.org/10.48550/arXiv.2310.07875>.

ELMAN, J. L. Finding structure in time. *Cognitive Science*, Wiley Online Library, v. 14, n. 2, p. 179–211, 1990.

FEDOROV, P.; MIRONOV, A.; CHERNISHEV, G. Russian web tables: A public corpus of web tables for russian language based on wikipedia. *Lobachevskii Journal of Mathematics*, Springer, v. 44, n. 1, p. 111–122, 2023.

GAO, K. Y.; CALLAN, J. Scientific table search using keyword queries. *CoRR*, abs/1707.03423, 2017. Available at: <http://arxiv.org/abs/1707.03423>.

GAO, L.; MA, X.; LIN, J.; CALLAN, J. Precise zero-shot dense retrieval without relevance labels. *CoRR*, abs/2212.10496, 2022. Available at: <https://doi.org/10.48550/arXiv.2212.10496>.

GAVRILOV, D.; KALAIDIN, P.; MALYKH, V. Self-attentive model for headline generation. In: *Proceedings of the 41st European Conference on Information Retrieval Research*. Cologne, Germany: Springer, 2019. (Lecture Notes in Computer Science, v. 11438), p. 87–93.

GLASS, M.; CANIM, M.; GLIOZZO, A.; CHEMMENGATH, S.; CHAKRAVARTI, R.; SIL, A.; PAN, F.; BHARADWAJ, S.; FAUCEGLIA, N. R. Capturing row and column semantics in transformer based question answering over tables. *CoRR*, abs/2104.08303, 2021. Available at: <https://arxiv.org/abs/2104.08303>.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep learning*. Cambridge, Massachusetts: The Mit Press, 2016.

GOVINDARAJU, V.; ZHANG, C.; RÉ, C. Understanding tables in context using standard NLP toolkits. In: *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Sofia, Bulgaria: The Association for Computer Linguistics, 2013. p. 658–664.

GU, X.; MAO, Y.; HAN, J.; LIU, J.; WU, Y.; YU, C.; FINNIE, D.; YU, H.; ZHAI, J.; ZUKOSKI, N. Generating representative headlines for news stories. In: *Proceedings of the Web Conference 2020*. Taipei, Taiwan: ACM, 2020. p. 1773–1784.

GUO, J.; FAN, Y.; AI, Q.; CROFT, W. B. A deep relevance matching model for ad-hoc retrieval. In: *Proceedings of the 25th ACM International Conference on Information and Knowledge Management*. Indianapolis, USA: ACM, 2016. p. 55–64.

GUO, J.; FAN, Y.; JI, X.; CHENG, X. Matchzoo: A learning, practicing, and developing system for neural text matching. In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Paris,France: ACM, 2019. p. 1297–1300.

HAYKIN, S. *Neural networks: a comprehensive foundation*. Hoboken, New Jersey: Prentice Hall, 1998.

HOCHREITER, S.; SCHMIDHUBER, J. Long short-term memory. *Neural Computation*, MIT Press, v. 9, n. 8, p. 1735–1780, 1997.

HOLTZMAN, A.; BUYS, J.; DU, L.; FORBES, M.; CHOI, Y. The curious case of neural text degeneration. In: *Proceedings of the 8th International Conference on Learning Representations*. Addis Ababa, Ethiopia: OpenReview.net, 2020.

HU, B.; LU, Z.; LI, H.; CHEN, Q. Convolutional neural network architectures for matching natural language sentences. *CoRR*, abs/1503.03244, 2015. Available at: <http://arxiv.org/abs/1503.03244>.

HUANG, P.; HE, X.; GAO, J.; DENG, L.; ACERO, A.; HECK, L. P. Learning deep structured semantic models for web search using clickthrough data. In: *Proceedings of the 22nd ACM International Conference on Information and Knowledge Management*. San Francisco, California: ACM, 2013. p. 2333–2338.

JÄRVELIN, K.; KEKÄLÄINEN, J. Cumulated gain-based evaluation of IR techniques. *ACM Transactions on Information Systems.*, v. 20, n. 4, p. 422–446, 2002.

JERONYMO, V.; BONIFACIO, L. H.; ABONIZIO, H.; FADAEE, M.; LOTUFO, R. de A.; ZAVREL, J.; NOGUEIRA, R. F. Inpars-v2: Large language models as efficient dataset generators for information retrieval. *CoRR*, abs/2301.01820, 2023. Available at: <https://doi.org/10.48550/arXiv.2301.01820>.

KACPRZAK, E.; KOESTEN, L.; IBÁÑEZ, L.; BLOUNT, T.; TENNISON, J.; SIMPERL, E. Characterising dataset search - an analysis of search logs and data requests. *Journal of Web Semantics*, v. 55, p. 37–55, 2019.

KACPRZAK, E.; KOESTEN, L.; TENNISON, J.; SIMPERL, E. Characterising dataset search queries. In: CHAMPIN, P.; GANDON, F.; LALMAS, M.; IPEIROTIS, P. G. (Ed.). *Proceedings of the The Web Conference 2018*. Lyon , France: ACM, 2018. p. 1485–1488.

KACPRZAK, E.; KOESTEN, L. M.; IBÁÑEZ, L.; SIMPERL, E.; TENNISON, J. A query log analysis of dataset search. In: CABOT, J.; VIRGILIO, R. D.; TORLONE, R. (Ed.). *Proceedings of the 17th International Conference on Web Engineering*. Rome, Italy: Springer, 2017. (Lecture Notes in Computer Science, v. 10360), p. 429–436.

KARPUKHIN, V.; OGUZ, B.; MIN, S.; LEWIS, P. S. H.; WU, L.; EDUNOV, S.; CHEN, D.; YIH, W. Dense passage retrieval for open-domain question answering. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. Virtual Event: Association for Computational Linguistics, 2020. p. 6769–6781.

KATO, M. P.; OHSHIMA, H.; LIU, Y.; CHEN, H. A test collection for ad-hoc dataset retrieval. In: DIAZ, F.; SHAH, C.; SUEL, T.; CASTELLS, P.; JONES, R.; SAKAI, T. (Ed.). *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Virtual Event: ACM, 2021. p. 2450–2456.

KATO, M. P.; OHSHIMA, H.; LIU, Y.-H.; CHEN, H.-L.; NAKANO, Y. Overview of the ntcir-16 data search 2 task. In: *Proceedings of the 16th NTCIR Conference on Evaluation of Information Access Technologies*. Tokyo, Japan: National Institute of Informatics, 2022. p. 12–18.

KIM, D. H.; HOQUE, E.; KIM, J.; AGRAWALA, M. Facilitating document reading by linking text and tables. In: *Proceedings of the 31st Annual ACM Symposium on User Interface Software and Technology*. Berlin, Germany: ACM, 2018. p. 423–434.

KOEHN, P.; OCH, F. J.; MARCU, D. Statistical phrase-based translation. In: HEARST, M. A.; OSTENDORF, M. (Ed.). *Proceedings of the 2003 Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*. Edmonton, Canada: The Association for Computational Linguistics, 2003. p. 48–54.

KURLAND, O. The cluster hypothesis in information retrieval. In: *Proceedings of the 36th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Dublin, Ireland: ACM, 2013. p. 1126.

LE, Q. V.; MIKOLOV, T. Distributed representations of sentences and documents. In: *Proceedings of the 31th International Conference on Machine Learning*. Beijing, China: JMLR.org, 2014. v. 32, p. 1188–1196.

LEE, K.; CHANG, M.; TOUTANOVA, K. Latent retrieval for weakly supervised open domain question answering. In: KORHONEN, A.; TRAUM, D. R.; MÀRQUEZ, L. (Ed.). *Proceedings of the 57th Conference of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 2019. p. 6086–6096.

LEES, A. W.; YU, C.; KORN, F.; SILVA, L. de S.; BARBOSA, L.; WU, Y. W. Collocating structured web tables with news articles. In: *Proceedings of the 1st International Workshop on News Recommendation and Intelligence co-located with The Web Conference 2021*. Ljubljana, Slovenia: ACM, 2021. p. 393–401.

LI, J.; DOU, Z.; ZHU, Y.; ZUO, X.; WEN, J. Deep cross-platform product matching in e-commerce. *Information Retrieval Journal*, v. 23, n. 2, p. 136–158, 2020.

LIMAYE, G.; SARAWAGI, S.; CHAKRABARTI, S. Annotating and searching web tables using entities, types and relationships. *Proceedings of the VLDB Endowment*, v. 3, n. 1, p. 1338–1347, 2010.

LIN, T.; CHEN, Q.; CHENG, G.; SOYLU, A.; ELL, B.; ZHAO, R.; SHI, Q.; WANG, X.; GU, Y.; KHARLAMOV, E. ACORDAR: A test collection for ad hoc content-based (RDF) dataset retrieval. In: AMIGÓ, E.; CASTELLS, P.; GONZALO, J.; CARTERETTE, B.; CULPEPPER, J. S.; KAZAI, G. (Ed.). *Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Madrid, Spain: ACM, 2022. p. 2981–2991.

LIN, T.; WANG, Y.; LIU, X.; QIU, X. A survey of transformers. *AI Open*, v. 3, p. 111–132, 2022.

LIU, T. Learning to rank for information retrieval. *Foundations and Trends in Information Retrieval.*, v. 3, n. 3, p. 225–331, 2009.

LIU, T.; ZHANG, X.; ZHANG, Z.; WANG, Y.; LI, Q.; ZHANG, S.; LIU, T. Enhancing table retrieval with dual graph representations. In: *Proceedings of the Machine Learning and Knowledge Discovery in Databases: Research Track*. Turin, Italy: Springer, 2023. (Lecture Notes in Computer Science, v. 14172), p. 107–123.

LIU, Y.; BAI, K.; MITRA, P.; GILES, C. L. Tablerank: A ranking algorithm for table search and retrieval. In: *Proceedings of the 22nd AAAI Conference on Artificial Intelligence*. Vancouver, Canada: AAAI, 2007. p. 317–322.

LIU, Y.; BAI, K.; MITRA, P.; GILES, C. L. Tableseer: automatic table metadata extraction and searching in digital libraries. In: *Proceedings of the 7th ACM/IEEE Joint Conference on Digital Libraries*. Vancouver,Canada: ACM, 2007. p. 91–100.

MAITY, S. K.; PANIGRAHI, A.; GHOSH, S.; BANERJEE, A.; GOYAL, P.; MUKHERJEE, A. Deeptagrec: A content-cum-user based tag recommendation framework for stack overflow. In: SPRINGER. *Proceedings of the 41st European Conference on Information Retrieval*. Cologne, Germany, 2019. p. 125–131.

MARZOCCHI, M.; CREMASCHI, M.; POZZI, R.; AVOGADRO, R.; PALMONARI, M. Mammotab: A giant and comprehensive dataset for semantic table interpretation. In: *Proceedings of the Semantic Web Challenge on Tabular Data to Knowledge Graph Matching*. Virtual Conference: CEUR-WS.org, 2022. (CEUR Workshop Proceedings, v. 3320), p. 28–33.

METZLER, D.; CROFT, W. B. Combining the language model and inference network approaches to retrieval. *Information Processing & Management*, v. 40, n. 5, p. 735–750, 2004.

METZLER, D.; CROFT, W. B. A markov random field model for term dependencies. In: *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Salvador, Brazil: ACM, 2005. p. 472–479.

MIKOLOV, T.; CHEN, K.; CORRADO, G.; DEAN, J. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781, 2013. Available at: <http://arxiv.org/abs/1301.3781>.

MIKOLOV, T.; SUTSKEVER, I.; CHEN, K.; CORRADO, G. S.; DEAN, J. Distributed representations of words and phrases and their compositionality. In: *Proceedings of the 27th Annual Conference on Neural Information Processing Systems*. Lake Tahoe, Nevada: Curran Associates, 2013. p. 3111–3119.

MITRA, B.; CRASWELL, N. An introduction to neural information retrieval. *Foundations and Trends in Information Retrieval*, v. 13, n. 1, p. 1–126, 2018.

MITRA, B.; DIAZ, F.; CRASWELL, N. Learning to match using local and distributed representations of text for web search. In: *Proceedings of the 26th International Conference on World Wide Web*. Perth, Australia: ACM, 2017. p. 1291–1299.

NALLAPATI, R.; ZHOU, B.; SANTOS, C. N. dos; GÜLÇEHRE, Ç.; XIANG, B. Abstractive text summarization using sequence-to-sequence rnns and beyond. In: *Proceedings of the 20th Conference on Computational Natural Language Learning*. Berlin, Germany: ACL, 2016. p. 280–290.

NOGUEIRA, R.; CHO, K. Passage re-ranking with BERT. *CoRR*, abs/1901.04085, 2019. Available at: <http://arxiv.org/abs/1901.04085>.

NOGUEIRA, R. F.; YANG, W.; LIN, J.; CHO, K. Document expansion by query prediction. *CoRR*, abs/1904.08375, 2019. Available at: <http://arxiv.org/abs/1904.08375>.

OMIDVAR, A.; AN, A. Learning to generate popular headlines. *IEEE Access*, v. 11, p. 60904–60914, 2023.

OUYANG, L.; WU, J.; JIANG, X.; ALMEIDA, D.; WAINWRIGHT, C. L.; MISHKIN, P.; ZHANG, C.; AGARWAL, S.; SLAMA, K.; RAY, A.; SCHULMAN, J.; HILTON, J.; KELTON, F.; MILLER, L.; SIMENS, M.; ASKELL, A.; WELINDER, P.; CHRISTIANO, P. F.; LEIKE, J.; LOWE, R. Training language models to follow instructions with human feedback. *CoRR*, abs/2203.02155, 2022. Available at: <https://doi.org/10.48550/arXiv.2203.02155>.

OVALLE, J. E. A.; SOLORIO, T.; MONTES-Y-GÓMEZ, M.; GONZÁLEZ, F. A. Gated multimodal units for information fusion. In: *Proceedings of the Workshop of the 5th International Conference on Learning Representations*. Toulon, France: OpenReview.net, 2017.

PANG, L.; LAN, Y.; GUO, J.; XU, J.; WAN, S.; CHENG, X. Text matching as image recognition. In: *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. Phoenix, USA: AAAI, 2016. p. 2793–2799.

PIMPLIKAR, R.; SARAWAGI, S. Answering table queries on the web using column keywords. *Proceedings of the VLDB Endowment*, v. 5, n. 10, p. 908–919, 2012.

PONTE, J. M.; CROFT, W. B. A language modeling approach to information retrieval. In: *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. Melbourne, Australia: ACM, 1998. p. 275–281.

PYREDDY, P.; CROFT, W. B. TINTIN: A system for retrieval in text tables. In: *Proceedings of the 2nd ACM International Conference on Digital Libraries*. Philadelphia, Pennsylvania: ACM, 1997. p. 193–200.

RAFFEL, C.; SHAZEER, N.; ROBERTS, A.; LEE, K.; NARANG, S.; MATENA, M.; ZHOU, Y.; LI, W.; LIU, P. J. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, JMLRORG, v. 21, n. 1, p. 5485–5551, 2020.

REIMERS, N.; GUREVYCH, I. Sentence-bert: Sentence embeddings using siamese bert-networks. In: INUI, K.; JIANG, J.; NG, V.; WAN, X. (Ed.). *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*. Hong Kong, China: Association for Computational Linguistics, 2019. p. 3980–3990.

RISTOSKI, P.; PAULHEIM, H. Rdf2vec: RDF graph embeddings for data mining. In: *Proceedings of the 15th International Semantic Web Conference*. Kobe, Japan: Springer, 2016. (Lecture Notes in Computer Science, v. 9981), p. 498–514.

ROBERTSON, S. E.; ZARAGOZA, H. The probabilistic relevance framework: BM25 and beyond. *Foundations and Trends in Information Retrieval*, v. 3, n. 4, p. 333–389, 2009.

ROITMAN, H.; MASS, Y. Utilizing passages in fusion-based document retrieval. In: *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*. Santa Clara, California: ACM, 2019. p. 59–66.

RONG, X. word2vec parameter learning explained. *CoRR*, abs/1411.2738, 2014. Available at: <http://arxiv.org/abs/1411.2738>.

RUSH, A. M.; CHOPRA, S.; WESTON, J. A neural attention model for abstractive sentence summarization. In: *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*. Lisbon, Portugal: The Association for Computational Linguistics, 2015. p. 379–389.

SAAD-FALCON, J.; KHATTAB, O.; SANTHANAM, K.; FLORIAN, R.; FRANZ, M.; ROUKOS, S.; SIL, A.; SULTAN, M. A.; POTTS, C. UDAPDR: unsupervised domain adaptation via LLM prompting and distillation of rerankers. *CoRR*, abs/2303.00807, 2023. Available at: <https://doi.org/10.48550/arXiv.2303.00807>.

SACHAN, D. S.; LEWIS, M.; JOSHI, M.; AGHAJANYAN, A.; YIH, W.; PINEAU, J.; ZETTLEMOYER, L. Improving passage retrieval with zero-shot question generation. In: GOLDBERG, Y.; KOZAREVA, Z.; ZHANG, Y. (Ed.). *Proceedings of the 2022 Conference on Empirical Methods in Natural Language Processing*. Abu Dhabi, United Arab Emirates: Association for Computational Linguistics, 2022. p. 3781–3797.

SALTON, G.; YANG, C.-S. On the specification of term values in automatic indexing. *Journal of Documentation*, v. 29, n. 4, p. 351–372, 1973.

SANH, V.; DEBUT, L.; CHAUMOND, J.; WOLF, T. Distilbert, a distilled version of BERT: smaller, faster, cheaper and lighter. *CoRR*, abs/1910.01108, 2019. Available at: <http://arxiv.org/abs/1910.01108>.

SANH, V.; WEBSON, A.; RAFFEL, C.; BACH, S. H.; SUTAWIKA, L.; ALYAFEAI, Z.; CHAFFIN, A.; STIEGLER, A.; RAJA, A.; DEY, M.; BARI, M. S.; XU, C.; THAKKER, U.; SHARMA, S. S.; SZCZECHLA, E.; KIM, T.; CHHABLANI, G.; NAYAK, N. V.; DATTA, D.; CHANG, J.; JIANG, M. T.; WANG, H.; MANICA, M.; SHEN, S.; YONG, Z. X.; PANDEY, H.; BAWDEN, R.; WANG, T.; NEERAJ, T.; ROZEN, J.; SHARMA, A.; SANTILLI, A.; FÉVRY, T.; FRIES, J. A.; TEEHAN, R.; SCAO, T. L.; BIDERMAN, S.; GAO, L.; WOLF, T.; RUSH, A. M. Multitask prompted training enables zero-shot task generalization. In: *Proceedings of the 10th International Conference on Learning Representations*. Virtual Event: OpenReview.net, 2022.

SANTOS, C. N. dos; BARBOSA, L.; BOGDANOVA, D.; ZADROZNY, B. Learning hybrid representations to retrieve semantically equivalent questions. In: *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation*

of Natural Language Processing. Beijing, China: The Association for Computer Linguistics, 2015. p. 694–699.

SANTOS, C. N. dos; MA, X.; NALLAPATI, R.; HUANG, Z.; XIANG, B. Beyond [CLS] through ranking by generation. In: WEBBER, B.; COHN, T.; HE, Y.; LIU, Y. (Ed.). *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing*. [S.l.]: Association for Computational Linguistics, 2020. p. 1722–1727.

SANTOSH, T. Y. S. S.; SAHA, A.; GANGULY, N. MVL: multi-view learning for news recommendation. In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Virtual Event: ACM, 2020. p. 1873–1876.

SCHÜTZE, H.; MANNING, C. D.; RAGHAVAN, P. *Introduction to information retrieval*. Cambridge, United Kingdom: Cambridge University Press, 2008.

SHEN, Y.; HE, X.; GAO, J.; DENG, L.; MESNIL, G. A latent semantic model with convolutional-pooling structure for information retrieval. In: *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management*. Shanghai, China: ACM, 2014. p. 101–110.

SHRAGA, R.; ROITMAN, H.; FEIGENBLAT, G.; CANIM, M. Ad hoc table retrieval using intrinsic and extrinsic similarities. In: *Proceedings of The Web Conference 2020*. Taipei, Taiwan: ACM, 2020. p. 2479–2485.

SHRAGA, R.; ROITMAN, H.; FEIGENBLAT, G.; WEINER, B. Projection-based relevance model for table retrieval. In: *Proceedings of The Web Conference 2020*. Taipei, Taiwan: ACM, 2020. p. 28–29.

SHRAGA, R.; ROITMAN, H.; FEIGENBLAT, G.; CANIM, M. Web table retrieval using multimodal deep learning. In: *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Virtual Event: ACM, 2020. p. 1399–1408.

SILVA, L.; BARBOSA, L.; CASTELO, S.; ZHANG, H.; SANTOS, A.; FREIRE, J. Nyucin at the ntcir-16 dataset search 2 task. In: *Proceedings of the 16th NTCIR Conference on Evaluation of Information Access Technologies*. Tokyo, Japan: National Institute of Informatics, 2022. p. 38–45.

SMIRNOVA, A.; CUDRÉ-MAUROUX, P. Relation extraction using distant supervision: A survey. *ACM Computing Surveys*, v. 51, n. 5, p. 106:1–106:35, 2019.

SONG, K.; TAN, X.; QIN, T.; LU, J.; LIU, T. Mpnet: Masked and permuted pre-training for language understanding. In: *Proceedings of the 33rd Advances in Neural Information Processing Systems*. Virtual Event: Curran Associates, 2020. p. 16857–16867.

SUN, C.; QIU, X.; XU, Y.; HUANG, X. How to fine-tune bert for text classification? In: SPRINGER. *Proceedings of the 19th China National Conference on Chinese Computational Linguistics*. Hainan, China, 2019. p. 194–206.

SUN, H.; MA, H.; HE, X.; YIH, W.; SU, Y.; YAN, X. Table cell search for question answering. In: *Proceedings of the 25th International Conference on World Wide Web*. Montreal, Canada: ACM, 2016. p. 771–782.

SUN, Y.; YAN, Z.; TANG, D.; DUAN, N.; QIN, B. Content-based table retrieval for web queries. *Neurocomputing*, v. 349, p. 183–189, 2019.

THAKUR, N.; REIMERS, N.; DAXENBERGER, J.; GUREVYCH, I. Augmented SBERT: data augmentation method for improving bi-encoders for pairwise sentence scoring tasks. In: *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. Virtual Event: Association for Computational Linguistics, 2021. p. 296–310.

TRABELSI, M.; CHEN, Z.; ZHANG, S.; DAVISON, B. D.; HEFLIN, J. Strubert: Structure-aware BERT for table search and matching. In: *Proceedings of the The Web Conference 2022*. Lyon, France: ACM, 2022. p. 442–451.

TRABELSI, M.; CHEN, Z.; ZHANG, S.; DAVISON, B. D.; HEFLIN, J. Strubert: Structure-aware BERT for table search and matching. In: *Proceedings of the The Web Conference 2022*. [S.l.]: ACM, 2022. p. 442–451.

TRABELSI, M.; DAVISON, B. D.; HEFLIN, J. Improved table retrieval using multiple context embeddings for attributes. In: *Proceedings of the 2019 IEEE International Conference on Big Data*. Los Angeles, California: IEEE, 2019. p. 1238–1244.

VASWANI, A.; SHAZEER, N.; PARMAR, N.; USZKOREIT, J.; JONES, L.; GOMEZ, A. N.; KAISER, L.; POLOSUKHIN, I. Attention is all you need. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. Long Beach, California: Curran Associates, 2017. p. 5998–6008.

VENETIS, P.; HALEVY, A. Y.; MADHAVAN, J.; PASCA, M.; SHEN, W.; WU, F.; MIAO, G.; WU, C. Recovering semantics of tables on the web. *Proceedings of the VLDB Endowment*, v. 4, n. 9, p. 528–538, 2011.

WAN, S.; LAN, Y.; GUO, J.; XU, J.; PANG, L.; CHENG, X. A deep architecture for semantic matching with multiple positional sentence representations. In: *Proceedings of the 30th AAAI Conference on Artificial Intelligence*. Phoenix, Arizona: AAAI, 2016. p. 2835–2841.

WU, Y.; SCHUSTER, M.; CHEN, Z.; LE, Q. V.; NOROUZI, M.; MACHEREY, W.; KRIKUN, M.; CAO, Y.; GAO, Q.; MACHEREY, K.; KLINGNER, J.; SHAH, A.; JOHNSON, M.; LIU, X.; KAISER, L.; GOUWS, S.; KATO, Y.; KUDO, T.; KAZAWA, H.; STEVENS, K.; KURIAN, G.; PATIL, N.; WANG, W.; YOUNG, C.; SMITH, J.; RIESA, J.; RUDNICK, A.; VINYALS, O.; CORRADO, G.; HUGHES, M.; DEAN, J. Google's neural machine translation system: Bridging the gap between human and machine translation. *CoRR*, abs/1609.08144, 2016. Available at: <http://arxiv.org/abs/1609.08144>.

XIONG, C.; DAI, Z.; CALLAN, J.; LIU, Z.; POWER, R. End-to-end neural ad-hoc ranking with kernel pooling. In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Shinjuku, Japan: ACM, 2017. p. 55–64.

XIONG, C.; ZHONG, V.; SOCHER, R. Dynamic coattention networks for question answering. In: *Proceedings of the 5th International Conference on Learning Representations*. Toulon, France: OpenReview.net, 2017. p. 1–14.

ZHANG, L.; ZHANG, S.; BALOG, K. Table2vec: Neural word and entity embeddings for table population and retrieval. In: *Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval*. Paris, France: ACM, 2019. p. 1029–1032.

ZHANG, S.; BALOG, K. Entitables: Smart assistance for entity-focused tables. In: *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Shinjuku, Tokyo, Japan: ACM, 2017. p. 255–264.

ZHANG, S.; BALOG, K. Ad hoc table retrieval using semantic similarity. In: *Proceedings of the 2018 World Wide Web Conference*. Lyon, France: ACM, 2018. p. 1553–1562.

ZHANG, S.; BALOG, K. Web table extraction, retrieval, and augmentation: A survey. *ACM Transactions on Intelligent Systems and Technology*, v. 11, n. 2, p. 13:1–13:35, 2020.

ZHANG, T.; LADHAK, F.; DURMUS, E.; LIANG, P.; MCKEOWN, K. R.; HASHIMOTO, T. B. Benchmarking large language models for news summarization. *CoRR*, abs/2301.13848, 2023. Available at: <https://doi.org/10.48550/arXiv.2301.13848>.

ZHU, M. Recall, precision and average precision. *Department of Statistics and Actuarial Science, University of Waterloo, Waterloo*, v. 2, n. 30, p. 6, 2004.

ZHU, M.; AHUJA, A.; WEI, W.; REDDY, C. K. A hierarchical attention retrieval model for healthcare question answering. In: *Proceedings of the Web Conference 2019*. San Francisco, California: ACM, 2019. p. 2472–2482.

APPENDIX A – HYPERPARAMETER OPTIMIZATION

A.1 NTCIR DATASET

Table 23 – Hyperparameter optimization for docTTTTTquery model in NTCIR dataset. We evaluate Top-P, Top-K, Temperature, Beams and #Tokens. Bold values represent the best ones for each hyperparameter according to the similarity metric (Trial 13). We perform 20 trials in this setup.

Trial	Top-P	Top-K	Temperature	Beams	#Tokens	Similarity
0	0.99753571532049	19	0.73199394181140	2	49	0.25060728192329
1	0.95290418060841	8	0.86617614577493	2	83	0.24193036556243
2	0.99849549260809	2	0.83244264080042	1	51	0.29403916001319
3	0.96521211214797	10	0.52475643163223	2	57	0.23190517723560
4	0.95697469303260	31	0.29214464853521	2	67	0.22500579059123
5	0.95998368910791	40	0.51423443841361	2	42	0.22221367061138
6	0.95852620618436	31	0.06505159298527	3	98	0.22191141545772
7	0.96523068845866	41	0.09767211400638	3	66	0.22303807735443
8	0.97475884550556	7	0.03438852111521	3	55	0.21411322057247
9	0.96558555380447	34	0.52006802117781	2	51	0.23090267181396
10	0.99663576416884	20	0.96953711078786	1	79	0.32165956497192
11	0.99922201645123	20	0.97981011002056	1	79	0.31123277544975
12	0.98878591856443	20	0.9735908581868	1	81	0.32813856005668
13	**0.98814616851281**	**22**	**0.99893261251177**	**1**	**86**	**0.33506843447685**
14	0.98690579810967	25	0.69395651837033	1	92	0.30019602179527
15	0.98696281077438	49	0.67873966640578	1	87	0.32012477517127
16	0.98619553658070	17	0.84137966566886	1	75	0.31570756435394
17	0.98065788011792	13	0.31095030817347	1	100	0.26966729760169
18	0.99236676046105	25	0.98291786698820	1	90	0.32392159104347
19	0.97808467058280	14	0.77157688751253	1	73	0.31495064496994

Source: Created by the author

A.2 ACORDAR DATASET

Table 24 – Hyperparameter optimization for docTTTTTquery model in ACORDAR dataset. We evaluate Top-P, Top-K, Temperature, Beams and #Tokens. Bold values represent the best ones for each hyperparameter according to the similarity metric (Trial 18). We perform 20 trials in this setup.

Trial	Top-P	Top-K	Temperature	Beans	#Tokens	Similarity
0	0.99753571532049	19	0.73199394181140	2	49	0.43005654215812
1	0.95290418060841	8	0.86617614577493	2	83	0.42939409613609
2	0.99849549260809	2	0.83244264080042	1	51	0.48153457045555
3	0.96521211214797	10	0.52475643163223	2	57	0.40309613943099
4	0.95697469303260	31	0.29214464853521	2	67	0.38631853461265
5	0.9599836891079	40	0.51423443841361	2	42	0.40591791272163
6	0.95852620618436	31	0.06505159298527	3	98	0.37815451622009
7	0.96523068845866	41	0.09767211400638	3	66	0.39163118600845
8	0.97475884550556	7	0.03438852111521	3	55	0.34661749005317
9	0.96558555380447	34	0.52006802117781	2	51	0.40511742234230
10	0.99663576416884	20	0.96953711078786	1	79	0.50866651535034
11	0.99922201645123	20	0.97981011002056	1	79	0.51018273830413
12	0.98878591856443	20	0.9735908581868	1	81	0.50398689508438
13	0.98449452116748	22	0.99895442980611	1	81	0.51071953773498
14	0.98447683005603	25	0.69296014565948	1	92	0.49473536014556
15	0.98519207641372	49	0.67875158357884	1	72	0.49922794103622
16	0.97704234573880	17	0.84105152761915	1	89	0.50174766778945
17	0.99113201378887	13	0.30823095141294	1	75	0.45109614729881
18	**0.97758419308396**	**25**	**0.98291900905561**	**1**	**90**	**0.51310926675796**
19	0.97786530572585	24	0.75994976985145	1	98	0.50148248672485

Source: Created by the author

APPENDIX B – RECALL SETUPS FOR DATASET RETRIEVAL

B.1 RECALL SETUPS FOR DATASET RETRIEVAL

Table 25 – Distinct indexing setups for dataset retrieval. Bold values represent the best indexing configuration for NTCIR and ACORDAR benchmarks in terms of Recall@100, which we use as candidate datasets for re-ranking.

NTCIR Dataset		
#	**Indexing Attributes**	**Recall@100**
1	<Title, Description, Tags>	0.4907
2	<Title, Description>	0.4846
3	<Title, Tags>	0.3336
4	<Description, Tags>	0.4887
5	<Title>	0.2858
6	**<Description>**	**0.5250**
7	<Tags>	0.2435
ACORDAR Dataset		
#	**Indexing Attributes**	**Recall@100**
1	**<Title, Description, Tags>**	**0.7779**
2	<Title, Description>	0.7409
3	<Title, Tags>	0.7009
4	<Description, Tags>	0.7205
5	<Title>	0.5988
6	<Description>	0.6526
7	<Tags>	0.4576

Source: Created by the author