VINICIUS CLEVES DE OLIVEIRA CARMO

A framework for closed domain question answering systems in the low data regime

São Paulo
2023

VINICIUS CLEVES DE OLIVEIRA CARMO

A framework for closed domain question answering systems in the low data regime

Revised Version

Thesis submitted to the Polytechnic School of the Universidade de São Paulo in partial fulfillment of the requirements for the degree of Master of Science.

São Paulo
2023

Name: CARMO, Vinicius Cleves de Oliveira

Title: A framework for closed domain question answering systems in the low data regime.

Thesis submitted to the Polytechnic School of the Universidade de São Paulo in partial fulfillment of the requirements for the degree of Master of Science.

Approved in:

Jury members

Prof. PhD. _____

Institution: _____

Judgment: _____

Prof. PhD. _____

Institution: _____

Judgment: _____

Prof. PhD. _____

Institution: _____

Judgment: _____

VINICIUS CLEVES DE OLIVEIRA CARMO

A framework for closed domain question answering systems in the low data regime

Revised Version

Thesis submitted to the Polytechnic School of the Universidade de São Paulo in partial fulfillment of the requirements for the degree of Master of Science.

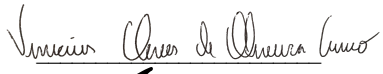
Área of Concentration:
Computer Engineering

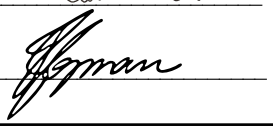
Supervisor:
Fabio Gagliardi Cozman

São Paulo
2023

Este exemplar foi revisado e corrigido em relação à versão original, sob responsabilidade única do autor e com a anuência de seu orientador.

São Paulo, 23 de Janeiro de 2023

Assinatura do autor: 

Assinatura do orientador: 

Catalogação-na-publicação

Carmo, Vinicius Cleves de Oliveira
A framework for closed domain question answering systems in the low data regime / V. C. O. Carmo -- versão corr. -- São Paulo, 2023.
64 p.

Dissertação (Mestrado) - Escola Politécnica da Universidade de São Paulo. Departamento de Engenharia de Computação e Sistemas Digitais.

1.Sistemas de questões e respostas 2.Recuperação da informação 3.Redes neurais 4.Aprendizado computacional I.Universidade de São Paulo. Escola Politécnica. Departamento de Engenharia de Computação e Sistemas Digitais II.t.

Para meus pais, Deanne e Osvaldo.

ACKNOWLEDGMENTS

I would like to thank my supervisor Prof. Dr. Fábio G. Cozman. His help, guidance and overall insights have made this work a very rich and inspiring experience for me.

I also gratefully acknowledge the support of ANP/PETROBRAS, Brazil (project 21721-6).

I would like to thank my research fellows at the Semantic Search Group at USP for the discussions that have certainly made this work better. I would also like to thank my friends at Turing USP, that made conversation about AI a daily and friendly thing.

Finalmente, eu agradeço a meu pai, Osvaldo, minha mãe, Deanne, meu irmão, Henrique, minha avó, Eliane, pelo apoio que me deram durante a realização desse trabalho; e a Pam, que me motivou e inspirou a perseguir esse mestrado.

RESUMO

CARMO, V. C. O. A framework for closed domain question answering systems in the low data regime / V. C. O. Carmo. 63p. Dissertação (Mestrado) – Escola Politécnica. Universidade de São Paulo - São Paulo, 2022.

Sistemas de resposta a perguntas (*Question Answering* – QA) que operam sobre conjuntos de documentos visam melhorar os sistemas tradicionais de recuperação de informações; enquanto estes recuperam documentos relevantes de uma base de documentos, aqueles também localizam e apresentam respostas diretas aos usuários. Melhorias recentes em QA tem sido baseadas em redes neurais, porém tais redes exigem grandes volumes de dados rotulados para treinamento. A maioria dos conjuntos de dados existentes contêm conhecimentos gerais e, embora existam alguns conjuntos de dados para domínios específicos (como biomedicina), na maioria dos domínios não há disponíveis conjuntos de dados rotulados ou fáceis de rotular. Isso cria um obstáculo para o desenvolvimento de sistemas de QA de domínio específico. Neste trabalho, propomos um esquema para desenvolvimento de sistemas de QA de domínio específico utilizando aprendizado não supervisionado, de modo a evitar os custos relacionados à rotulagem de grandes conjuntos de dados. Nossa contribuição tem duas formas. Primeiro, aplicamos a técnica de pré-treino adaptativo ao domínio para melhorar o desempenho fora do domínio em sistemas de compreensão de leitura e QA. Essa técnica atinge o estado da arte em dois conjuntos de dados de compreensão de leitura, e supera a performance de técnicas de adaptação de domínio no estado da arte na literatura por uma margem significativa: 2,3 em correspondência exata e 5.2 em F1-score no BioASQ. Em seguida, propomos um framework para QA em domínio específico em regime de escassez de dados. Para recuperação de documentos, aplicamos uma combinação do BM25 junto com um pipeline de processamento de texto personalizado. Descobrimos que, em um regime de escassez de dados, modelos estatísticos de recuperação de documentos superam os modelos neurais, conforme os dados no domínio desejado diferem dos dados utilizados durante o treinamento. Para a seleção de respostas, aplicamos um leitor neural treinado com a técnica de pré-treino adaptativo ao domínio para melhorar a generalização no domínio desejado. Também realizamos um estudo de caso aplicando o framework proposto ao domínio da engenharia oceânica.

Palavras-Chave – Sistemas de questões e respostas, recuperação de informação, redes neurais, aprendizado computacional.

ABSTRACT

CARMO, V. C. O. A framework for closed domain question answering systems in the low data regime / V. C. O. Carmo. 63p. Thesis (Master's Degree) – Polytechnic School. Universidade de São Paulo - São Paulo, 2022.

Question Answering (QA) systems that operate over textual databases aim at improving traditional information retrieval systems; while the latter recover a number of relevant documents from a document pool, the former can also find and present direct answers to users. Recent improvements on QA have been based on deep neural networks; such networks require large volumes of labeled data for training. Most existing datasets target general knowledge and, even though there are a few datasets for specific domains (such as biomedicine), for most domains there is no labeled, or easy to label, dataset available. This creates an obstacle for the development of domain-specific QA systems. We propose a framework for developing domain-specific QA systems by leveraging unsupervised learning so as to avoid the costs related to large scale dataset labeling. The contributions of this work are twofold. First, we apply domain-adaptive pretraining to improve out-of-domain performance of reading comprehension and question answering systems. This technique achieves state-of-the-art results on two Reading Comprehension datasets, and it exceeds the performance of state-of-the-art domain adaptation techniques in the literature by a significant margin: 2.3 exact match and 5.2 F1-score on BioASQ. Then, we propose a framework for domain-specific question answering in the low data regime. For document retrieval, we apply a combination of BM25 along with a custom text processing pipeline. We find that, in a low data setting, statistical document retrieval models outperform neural models as the data on the desired domain differ from the data used for training. For answer selection, we apply a neural reader trained with domain-adaptive pretraining to improve generalization on the desired domain. We also perform a case study by applying the proposed framework to the offshore engineering domain.

Keywords – Question answering systems, information retrieval, neural networks, machine learning.

LIST OF FIGURES

LIST OF TABLES

ACRONYMS

AP	Average Precision
BART	Bidirectional and Auto-Regressive Transformers
BERT	Bidirectional Encoder Representations from Transformers
BioASQ	Biomedical Semantic Indexing and Question Answering
BM25	Best Match 25
CLS	Classification token
CNN	Convolutional Neural Network
CPU	Central Processing Unit
DAPT	Domain-Adaptive PreTraining
DAPT-QA	Domain-Adaptive PreTraining for Question Answering
DPR	Dense Passage Retriever
ERNIE	Enhanced Representation through kNowledge IntEgration
FAISS	Facebook AI Similarity Search
GPT	Generative Pre-trained Transformer
GRU	Gated Recurrent Units
IDF	Inverse Document Frequency
IR	Information Retrieval
KB	Knowledge Base
LAT	Lexical Answer Type
LSTM	Long Short-Term Memory
MAP	Mean Average Precision
MLM	Masked Language Modeling
MLP	Multi-Layer Perceptron
MRQA	Machine Reading for Question Answering
MRR	Mean Reciprocal Ranking
NER	Named Entity Recognition
NLP	Natural Language Processing
NLQA	Natural Language Question Answering
NQ	Natural Questions
NSP	Next Sentence Prediction
OMAE	International Conference on Ocean, Offshore & Arctic Engineering
ORQA	Open Retrieval Question Answering
QA	Question Answering
RC	Reading Comprehension
REALM	Retrieval-Augmented Language Model Pre-Training
RNN	Recurrent Neural Network
RoBERTa	a Robustly optimized BERT approach
RR	Reciprocal Ranking
SEP	Separation token
SPARQL	SPARQL Protocol and RDF Query Language
SQL	Structured Query Language
SQuAD	Stanford Question Answering Dataset
TF	Term Frequency

TREC	Text REtrieval Conference
US	United States

CONTENTS

1 INTRODUCTION

There is an ever growing amount of text data produced around the world. This data can be mined so as to extract information that drives productivity, assist in decision-making, improve customer satisfaction and a wide range of other enterprise activities. Just as an example, a report from the McKinsey Global Institute in 2011 [2] highlighted the economic value that can be derived from Big Data, including text mining: according to this report, the US Health care sector could create more than $ 300 billion in value every year by using data to drive efficiency and quality; likewise, in the private sector, retailers, for example, could improve their operating margin by more than 60%.

Itto et al. [3] review the literature on text mining and Natural Language Processing (NLP) in the context of industrial applications. The authors present a set of challenges and desiderata related to such applications. Among the challenges are: (1) the heterogeneity of the data sources, with different formats, languages and text lengths; (2) the use of technical abbreviations and informal language; (3) the brevity of texts; (4) the imbalance of classes; (5) the lack of annotated data; (6) the need for "velocity" as some applications require fast responses or process large streams of text. The desiderata identified by them are: (1) flexibility to accommodate for the heterogeneity of the data sources; (2) robustness; (3) minimal supervision, taking into account the difficulty in obtaining annotated data and the costs involved in this process; (4) human intervention, where users who are experts in the application domain stay at the heart of the process; (5) easy of use, as users are not expected to be experts in text mining or NLP; (6) velocity; (7) extrinsic evaluation based on the effectiveness and usability of the application.

An example of a practical application of NLP in the industrial sector is the work by Tanguy at al. [4]. They describe four different systems, in increasing order of complexity, to assist aviation safety experts with text classification and information retrieval. The systems rely on a database of aviation incident reports. These reports are made of a free text section describing the incident together with metadata such as time, location and the equipment involved. The first proposed system is a learned classifier to assist with report

classification from an existing set of categories. The second system extracts topics from the data in order to find patterns that might not be covered by the existing categories. The third system retrieves reports that are similar to a given report, and displays them in a time line in order to show similar events over time. Finally, the fourth system builds an information retrieval system where the user is at the center or the process, working with the system in an interactive fashion.

Information Retrieval (IR) can greatly benefit from NLP. The goal of IR is to find, within a large document pool, a small set of documents that supply the information need expressed by a user query. Perhaps the most popular application of such information retrieval systems are web search engines, like Google,[1] Yahoo,[2] and Bing.[3] Such systems range from e-commerce product search to enterprise document retrieval.

Question Answering (QA) is an application of NLP that is directly related to IR. While IR looks for whole documents in response to an information need, QA is concerned with finding short and direct answers to user queries that are expressed in natural language and may be of different types. QA systems have the potential to provide better user experience and drive productivity in the search for information, when compared to IR systems.

QA systems can then be classified according to the source of knowledge they use to answer questions. They may be based on Knowledge Bases (KB), on text, or on mixed sources.

In KB-based QA systems the source of information is a Knowledge Base. In such structures, entities are represented as nodes and relations between them are represented as edges. A pair of entities connected by a relation forms an unit of knowledge. For example, in the unit $<e_1,\ r,\ e_2> = <$*Brasilia, capital of, Brazil* $>$, the entities $e_1 =$ *Brasilia* and $e_2 =$ *Brazil* are connected by the relation $r =$ *capital of*, representing the knowledge that Brasilia is Brazil's capital. Formal query languages, such as SPARQL, have been developed to interact with KBs. These query languages are very useful to experts interacting with KBs; however, they require programming ability and technical knowledge about the specific structures in the KBs to interact with them. Due to this, QA systems based on KBs have been designed that allow non-expert individuals to interact with KBs [5].

In text-based QA systems, the source of information are documents, for example,

[1] www.google.com
[2] search.yahoo.com
[3] www.bing.com

books, articles, tweets, social media posts. QA systems may also draw knowledge from multiple sources, mixing KBs and document pools; we classify the latter systems as mixed QA systems.

Text-based QA systems are often split in two modules. The first module, the Retriever, is an IR system. It recovers from the large document pool a set of documents it considers more relevant to the user query. The number of retrieved documents can be larger than what is expected for the user to read, as these documents will not be presented directly to the user, but rather to the second module in the pipeline. The second module, the Reader, generates or extracts the most likely answers based on the information on the retrieved documents and the query.

This work focuses on text-based QA, so, unless otherwise noted, when referring to QA systems we mean text-based QA systems.

Reading Comprehension (RC) is a NLP task where, given a passage and a question, the model must select a span in the passage that answers the question. This task can be seem as a close and simplified version of text-based QA. While in QA the system must process a collection of document with each query, in RC there is usually only one document to read along with each question.

QA systems can also be classified with respect to the application domain. A general-domain QA system is designed to answer questions about general knowledge. These systems can use, as information sources, documents from Wikipedia, for example. On domain-specific QA systems, on the other hand, the system is expected to answer questions about specific fields of knowledge, e.g., civil law, relativistic physics, offshore engineering.

Most research on domain-specific QA systems is based on linguistic approaches. Often, it requires custom pattern crafting or manual creation of ontologies. While these approaches may lead to high precision, they significantly reduce recall, as it is difficult for humans to identify all possible questions in advance. Another concern is that patterns or ontologies can be too specific for a domain, so patters designed for a domain may be useless for another domain [6].

The current state-of-the-art in general-domain QA research has moved towards the semantic representation of questions and documents using neural networks. In this approach, the system learns to answer questions based on examples, therefore avoiding the challenges and limitations on manual pattern crafting. The effectiveness of neural networks in capturing semantics and producing answers for questions has been demonstrated

on benchmarks like SQuAD [7] and Natural Questions [8].

Motivated by the improvements in general-domain QA, researchers have introduced machine learning approaches also for domain-specific QA. These are run in domains with significant available textual data, as is the case of biomedicine [9], as machine learning approaches requires a large amount of data for training.

The ability of neural networks to generalize — that is, to produce correct results on examples that were not seen during training — is closely related to how similar the new examples are from the ones in the training set. The performance of such models tends to degrade as new examples drift from the examples observed during training. For this reason, creating a good QA model for a specific domain is not as simple as training a model on any large scale dataset available, since the train/test mismatch can hurt considerably the performance of the final system.

In order to improve generalization, researchers have studied domain adaptation for QA systems [10]. In such configuration, models learn from datasets from one domain, e.g., the Wikipedia, and are evaluated on datasets from another domain, e.g., biomedicine. The goal is to make these models more robust to the diversity of questions and documents they may find, including when facing examples from different domains than the ones seem during training.

QA systems can also be designed to answer specific types of questions. Kolomiyets and Moens [11] classify questions in 10 major categories: factoid, list, definition, descriptive, opinion, hypothetical, causal, relationship, procedural, and confirmation questions. Factoid questions require a fact expressed in a document as an answer. These questions usually start with a wh-word (what, when, where, who). List questions require a list of entities or facts as answers. Definition questions ask for a definition of terms in the question. Descriptive questions ask for description of events, while opinion questions ask for opinions about entities or events. Hypothetical questions require information about hypothetical events, e.g, “what if” questions. Causal questions require an explanation for an event or artifact, e.g., “why” questions. Relationship questions ask about relations between entities. Procedural questions requires a list of instructions as answers. Confirmation questions require a yes/no answer. Much of the literature on text-based QA focuses on factoid questions, as their answers can be usually found as a text span in given documents.

1.1 Objectives

This work focuses on building a framework for Question Answering systems in closed domain applications. Such systems usually require a large amount of training data, pattern crafting or ontology creation efforts. We focus on factoid type questions, where the answer is a fact expressed in a text document. We propose a framework with relatively little crafting labor involved, where a QA system is split in two parts: the Retriever and the Reader.

For the Retriever, we have compared a state-of-the-art neural retriever trained on the general-domain against a strong statistical retriever with the appropriate pre-processing steps for indexing and retrieval. We recommend our best attained configuration for the Retriever. For the Reader, we propose a neural model trained with Domain-Adaptive Pretraining (DAPT) to improve in-domain performance when using out-of-domain annotated data and in-domain text data.

We developed and tested our ideas using a case study in offshore engineering. A complete QA system in that domain is as another key contribution of this work.

To summarize, our main contributions are: (1) we apply DAPT for QA and show that it achieve state-of-the-art performance on domain adaptation, outperforming more complicated techniques in the state-of-the-art; (2) we propose a framework for domain-specific question answering in a low data regime. Other key contributions include: (1) a QA System in the offshore engineering domain; (2) the Offshore QA Dataset, a factoid QA dataset on the offshore engineering domain and (3) OffshoreBert, a BERT pretrained model specialized on the offshore engineering domain.

1.2 Document structure

The remaining of this document is organized as follows: Chapter 2 reviews the literature and related work in linguistic and neural QA and domain adaptation for QA systems. Chapter 3 explains our methodology and presents the two main contributions of this work: the application of DAPT for domain adaptation of QA systems and our proposed framework for closed domain QA. Chapter 4 presents our experiments and results to validate our proposed domain adaptation approach.

Chapters 5, 6 and 7 are built around our case study on offshore engineering. Chapter 5 describes a small QA dataset that was built in this work and used to validate exper-

iments in our case study. Chapter 6 describes the experiments designed to understand the best configuration for the Retriever module and to justify our design choices. Chapter 7 presents OffshoreBERT, a language model specialized in offshore engineering (a by-product of DAPT), and a complete Reader module for that domain. Together, these Chapters represent the full implementation of the framework for closed domain QA proposed in this work applied on the offshore engineering domain. Finally, Chapter 8 presents our conclusions and comments on possible future work.

2 LITERATURE REVIEW

This chapter reviews the literature on text-based QA systems.

Section 2.1 reviews linguistic approaches to QA. Section 2.2 presents the reasoning and progress that has produced state-of-the-art semantic neural approaches. A few neural architectures are presented; in particular, the Transformer, the backbone for many state-of-the-art architectures on NLP tasks.

One key aspect regarding the use of machine learning is the need for training data. Section 2.3 discusses the automatic generation of QA pairs. Section 2.4 summarizes work that attempts to overcome the lack of training data in some domains by finding ways to use only unsupervised data.

2.1 Linguistic question answering

The linguistic approach to QA systems is based on question analysis using NLP tools such as Named Entities Recognition (NER) [12,13] and morphological and syntactic analysis [12,14,15]. Entities and relationships can be identified in questions and embedded in the search request sent to the structured knowledge bases to find answers. Query rewriting techniques can be applied to improve the requests sent to document search engines when dealing with unstructured sources [16]. The selected passages can also be analyzed with NLP tools and possible responses can be ranked based on frequency and suitability [12]. These systems have yet to deal with the issue of different words representing the same idea in questions, documents or labels in the database, a problem known as lexical gap. WordNet is a tool applied in this context [17], as it organizes words with similar meaning in groups called synsets, and organizes synsets in a hierarchy of concepts, thus enabling words to be matched at a semantic level [18].

For instance, Abdi et al. [14] created an ontology-based QA system for the Electricity and Electromagnetism domains. They use an ontology designed for these domains to

extract entities and relations from text and store this information on a database. They also collect a large number of questions based on the information provided by the ontology and extract patterns that are mapped to SQL queries and grouped in clusters with the same SQL query patterns. Upon receiving a new question, the system interprets the question according to the ontology and searches for its corresponding query cluster based on similarity. If a cluster is found, the SQL query is used to retrieve the answer.

The Natural Language Question Answering (NLQA) system [12] uses a set of NLP techniques along with a domain ontology to retrieve answers to questions. The system processes questions using semantic and syntactic analysis and employs the domain ontology to reformulate the input question into a query to a retrieval engine. This engine makes use of conceptual indexing that allows for the semantic role label to be indexed along with the corresponding word in the inverted index, accounting for the semantics in the retrieval process [19]. Once a set of documents are retrieved, they are syntactic and semantically analyzed and the answers are ranked based on semantic similarity or generated with help from the ontology.

Damiano et al. [13] uses NLP techniques to understand the information need in factoid questions and to formulate a query to an information retrieval engine. The system identifies named entities, question keywords (such as nouns, verbs and dates), answer types; then the system uses that information to formulate a query. For example, for the question

> When was the Colosseum built?

the corresponding query would be

> **LAT**: *Date*, **Named Entity**: *Colloseum*, **Verbs**: *built*,

where LAT stands for Lexical Answer Type. From the relevant documents retrieved, they extract passages and use answer type information, named entity matching, keyword overlapping, and local proximity between matching terms to filter and score the answer sentences.

2.2 Neural question answering

Recent work on NLP has explored semantic but numeric representations for words, sentences, and documents. In particular, *word embeddings* have been very popular.

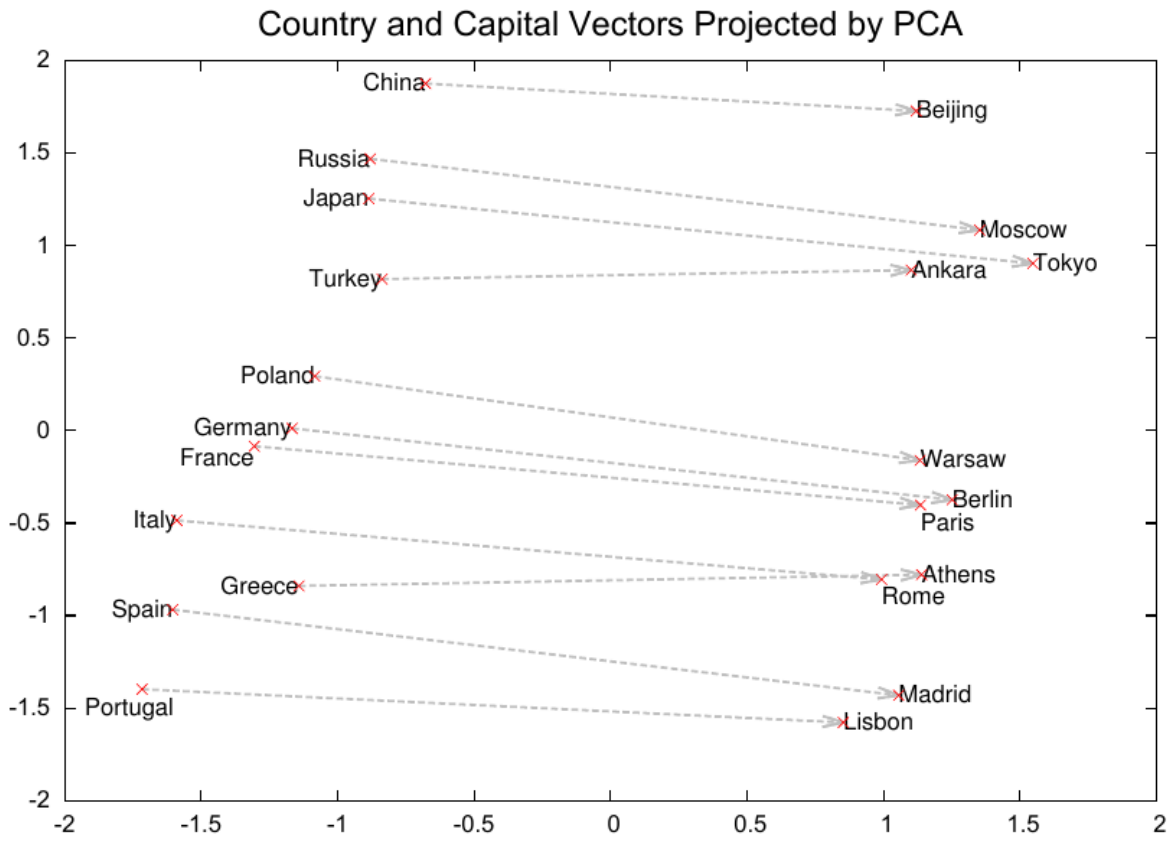


Figure 1: Two dimensional Principal Component Analysis projection of word-embeddings for countries and their capital cities.
Source: Ref . [20].

Word embeddings are vector representations of words, where each word is represented by a vector in a n-dimensional space that is learned based on word proximity in a text corpus [20]. Word embeddings can even be used to capture relationships between the meaning of words: for example, the operation

"Berlin" - "Germany" + "France"

may lead to the vector for "Paris" (Figure 1). The space in which these vectors are represented is often referred to as a *semantic space*. Representations for larger units of text can be built from the representation of words. For instance, a straightforward representation is to take the average of the words in a sentence [21]. Other approaches involve using Convolutional Neural Networks [22], Recursive Neural Networks [23], and Recurrent Neural Networks [24] to reduce the set of word vector in a sentence to a single vector representation.

In the neural approach to QA, instead of embedding linguistic knowledge into the system, the designers focus on building an architecture such that the system can learn

whatever is important to find answers to questions. In this setting, a key idea is to find a function that maps queries and its answers to similar locations in the semantic space, thus reducing the problem of finding the best answer to a query to the problem of finding closest vectors in the semantic space.

Tay et al. [21] take pre-trained word embeddings and use a feedforward layer to build a task specific representation for words. These representations are summed and projected into a unit ball, and similarities between questions and answers are calculated on a hyperbolic space. Qiu and Huang [22] use a Convolutional Neural Network (CNN) to build a tensor layer that matches question and answers. Lukovnikov et al. [24] use a RNN/GRU to get vector representations for words by starting from characters. These character-level representations are concatenated with word-level embeddings and fed to another RNN/GRU to build the final sentence representation. The character-level representation allows the network to deal with out-of-vocabulary words. Tay et al. [25] use multilayered RNN/LSTMs to encode the questions and answers from word vectors. These representations interact on a correlation layer followed by two fully connected layers. A bilinear similarity score and some hand-crafted features, like word overlap, are appended before the fully connected layers; the justification for the hand-crafted features is that such features are hard to be learned autonomously by the model.

Tran and Niedereée [26] apply RNN/BiLSTMs and attention layers to match question and answers (a RNN/BiLSTM consists of a pair of RNN/LSTM networks, one that goes forward on the sentence, and another one that goes backwards). The advantage of using RNN/BiLSTMs is that they can model forward and backward dependency on text. One drawback of RNNs is that they have difficulty in modeling long term dependency on text. On every interaction, RNNs "forgets" a little about the data they have processed. Attention layers, or attention mechanisms, can instead model these long term dependencies on text. There are several types of attention, e.g., dot-product, additive, bilinear, sequential, self (Ref. [26] offers a detailed description of the last four techniques).

Taking the dot-product attention as an example, let $P = \{\mathbf{p_1}, ..., \mathbf{p_N}\} \in \mathbb{R}^{N \times E}$ and $Q = \{\mathbf{q_1}, ..., \mathbf{q_M}\} \in \mathbb{R}^{M \times E}$ be two sequences of word vectors, where E is the embedding size. The word $\mathbf{p_i}$ generates the vector $\mathbf{v} \in \mathbb{R}^M$ and the attention vector $\mathbf{a} \in \mathbb{R}^M$:

$$\mathbf{v} = [\mathbf{p_i} \cdot \mathbf{q_1}, ..., \mathbf{p_i} \cdot \mathbf{q_M}], \tag{2.1}$$

$$\mathbf{a} = \text{softmax}(\mathbf{v}), \tag{2.2}$$

where $\cdot$ denotes the inner product. The new representation for $\mathbf{p_i}$ is result of words in Q

weighted by $\mathbf{a}$:

$$\mathbf{p_i^{'}} = \sum_j a_j \cdot \mathbf{q_j}. \tag{2.3}$$

The procedure above describes the process for a single token. By applying the same procedure to every token in P we get:

$$P' = \text{softmax}(P * Q^T) * Q, \tag{2.4}$$

where $*$ denotes matrix multiplication.

An important advance in neural networks for NLP has been the Transformer architecture [27]. It dropped the use of CNNs and RNNs in favor of an attention-based architecture. As attention mechanisms have no built-in ways to represent order, the Transformer introduces the concept of positional encoding: an array of values, learned or hard-coded, added to the regular word embeddings to represent word positions on a sentence. This migration from RNNs to attention mechanisms has also simplified the parallelization of computations.

Figure 2 shows the Transformer architecture. It has a modular encoder-decoder structure that is well suited to its original machine translation task. First the input is encoded by Nx blocks in the encoder. Then, the output is generated (one element at a time) in the decoder by going through Nx block of self-attention and attention over the output of the encoder.

Part of the Transformer-based model success can be attributed to its training with transfer learning [28]. The neural model is first subjected to a pretraining step, where it learns a language modeling task. Then, this model goes through a finetuning step, where it is transferred to another task with a small amount of training. Transfer learning had already shined in other areas of machine learning, such as computer vision, but it was not very successful in NLP until the Transformer architecture was proposed.

Transformer-based architectures have set new levels of performance in a variety of tasks. For instance, BERT (Bidirectional Encoder Representation from Transformer) [28] is a model equivalent to the Transformer encoder. Inputs to BERT are tokenized using subword units, and two special tokens are added to the sentence. The $[CLS]$ token is added as first token, and represents the beginning of a sequence; to allow BERT to represent a pair of sentences, the $[SEP]$ token is added at the end of the first sentence. BERT is pretrained simultaneously on the task of masked language modeling and next sentence prediction.

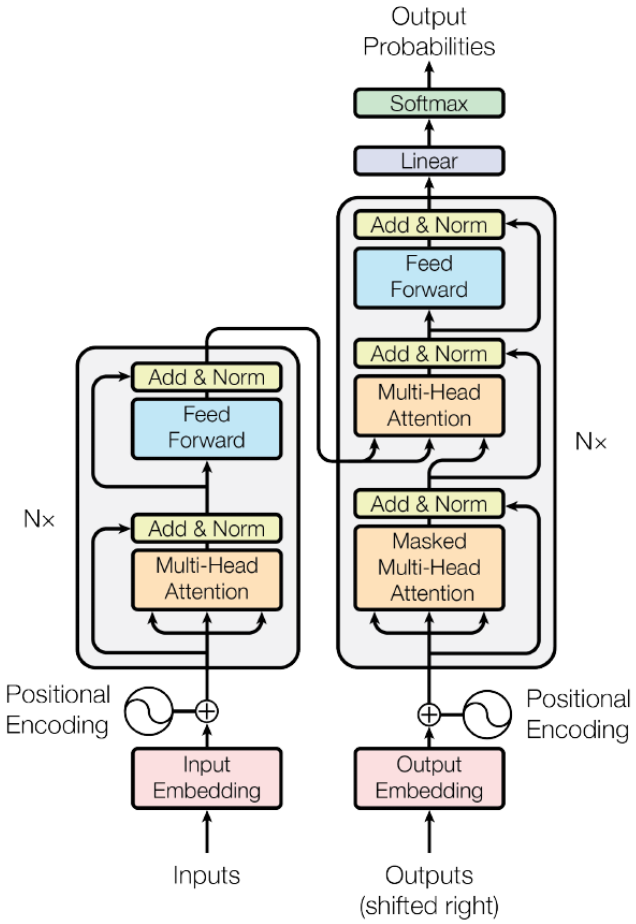


Figure 2: The Transformer architecture.

Source: Ref . [27].

In Masked Language Modeling (MLM), some tokens are masked from the input and are predicted by the language model. Next Sentence Prediction (NSP) is the task of predicting whether or not the second sentence follows the first one in the corpus. The classification for NSP is built from the BERT output for the $[CLS]$ token. While the MLM task captures the structure of language, the NSP task builds representations for relations between sentences. BERT established state-of-the-art results on eleven NLP tasks, including SQuAD v1.1 [7] and v2.0 [29] for question answering.

RoBERTa [30], a Robustly optimized BERT approach, introduces a series of changes on the BERT pretraining approach. Notably, RoBERTa: (1) drops on the use of the NSP task, (2) uses dynamic masking for the MLM objective, where different tokens are masked every time a sequence is fed to the model, (3)increases training batch sizes, (4) pretrains longer and (5) increases the diversity of data during training. In addition to the two corpora used in BERT pretraining, english Wikipedia and BookCorpus, RoBERTa is also pretrained with CC-News, containing 63 million English news articles, OpenWebText,

with documents extracted from URLs shared on Reddit with at least three upvotes, and Stories, containing texts to target common sense reasoning.

Another notable model is GPT-2 [31], a language model based on the Transformer decoder. The model is trained for next token prediction, making it a suitable model for natural language generation. GPT-2 has established new state-of-the-art results on several language modeling datasets on a zero-shot setting, where no fine-tuning is performed on the testing dataset.

The DPR Reader [32] is an example of a Transformer-based QA system. Given a small set of passages (up to 100 in the original article) and a question, DPR Reader ranks the passages in order of relevance and selects the answer spans inside them. It uses three feed-forward layers on top of a pretrained BERT model to score the relevance of each sentence and the probability of each token being the start or end of the answer span.

Embedding-based semantic representations on QA have been applied to the second part of the question answering pipeline, that is, to sentence or answer selection. The same semantic representations can be used in the first part, information retrieval, to encode the whole set of documents and to select them based on the proximity with the query. This retrieval strategy has been shown to improve question answering when applied on datasets such as Natural Questions (NQ) [8], for example, built from Google queries [33].

In particular, the Dense Passage Retriever (DPR) [32] splits documents in 100-word passages and uses a BERT model to encode the passages into vector representations. This representation is indexed using FAISS [34], an efficient indexer for dense vector spaces. At querying time, the question in natural language is encoded using another BERT model and the output from the $[CLS]$ token is matched by dot-product similarity with the passage vectors indexed in FAISS. On a server with Intel Xeon CPU E5-2698 v4 @ 2.20GHz and 512GB memory, DPR can process 995.0 questions per second, returning top 100 passages per question from 21 million indexed passages from Wikipedia.

REALM (Retrieval-Augmented Language Model Pre-Training) [35] jointly models the retrieval and answer extraction task of question answering in an end-to-end implementation. Similarly to DPR, REALM splits the documents in passages of 288 tokens. Let y be the prediction objective (masked token for pre-training and answer span for QA), x the input (masked sentence for pre-training and question for QA) and z a passage from a corpus $\mathcal{Z}$. Then the probability $p(y \mid x)$ is decomposed as

$$p(y \mid x) = \sum_{z \in \mathcal{Z}} p(y \mid z, x)\, p(z \mid x), \tag{2.5}$$

where $p(z \mid x)$ corresponds to the retriever and is modeled using two BERT models in a similar fashion to DPR, with the exception that the representations to questions and passages are linearly projected for dimensionality reduction. For pre-training, $p(y \mid z, x)$ is modeled using BERT:

$$p(y \mid z, x) = \prod_{j=1}^{J_x} p\left(y_j \mid z, x\right), \tag{2.6}$$

$$p\left(y_j \mid z, x\right) \propto \exp\left(w_j^\top \text{BERT}\left(x, z\right)\left[\text{MASK}\left(j\right)\right]\right), \tag{2.7}$$

where MASK(j) denote the position of the j^{th} masked token in the training example, BERT$(x, z)[k]$ denotes the BERT output for k^{th} token from the inputs (x, z), J_x is the number of masked token on x and w_j is the learned word embedding for token y_j. For QA fine-tuning, $p(y \mid z, x)$ is modeled using BERT and a MLP to classify the answer spans:

$$p(y \mid z, x) \propto \sum_{(s,e) \in S(z,y)} \exp\left(\text{MLP}\left(\left[\text{BERT}(x, z)[s]; \text{BERT}(x, z)[e]\right]\right)\right), \tag{2.8}$$

where $S(z, y)$ is the set of spans matching y in z represented by the start (s) and end (e) token positions.

Open Retrieval Question Answering (ORQA) [33] follows an approach similar to REALM to model the retriever and the reader with BERT. ORQA pre-training, though, is done on the Inverse Cloze Task, which consist in predicting the sentences that are close to another, and pre-trains only the retriever. During fine-tuning, the retriever passage encoder is frozen and only the retriever question encoder and reader are trained. Similar to REALM, documents are split in passages of 288 BERT tokens.

Yang et al. [36] examine whether the improvements brought by Neural IR models are effective or if they are just the result of weak baselines. They study results reported in the literature in the TREC 2004 Robust Track (Robust04) test collection from 2005 to 2018 and also add a strong BM25 with query expansion baseline. They observe that articles often report weak baselines and that the best result reported in the Robust04 was non-neural. Most Neural IR results were below the traditional BM25 algorithm with query expansion baseline and only one out of five neural reranking models was able to improve baseline performance. The authors acknowledge the limitations of their study as it considers only a single IR dataset; also neural IR models may benefit from other settings with much more data available, such as data from search logs, for example.

BM25 is one of the most often used similarity functions for IR models. It is the default similarity function both on Solr and Elasticsearch, arguably the two most popular

enterprise search engines. The intuition behind BM25 is that words that appears on a document more times (i.e., have a high term frequency) are more informative about the content of that document. Also, words that appears in many documents (i.e., have a low inverse document frequency) are not discriminative of the content of a particular document in relation to others in the document collection. Therefore, BM25 favors documents in the collection that have many terms from the query that are rare in the collection. Some variations of the BM25 formula can be found on the literature. The following is adapted from Robertson and Zaragoza [37]:

$$\text{BM25}_{\text{score}} = \sum_i w^{\text{BM25}}(q_i), \tag{2.9}$$

$$w^{\text{BM25}}(q_i) = w^{\text{TF}}(q_i) \cdot w^{\text{IDF}}(q_i), \tag{2.10}$$

$$w^{\text{TF}}(q_i) = \frac{tf(q_i)}{k_1 \left((1-b) + b\frac{dl}{avdl}\right)}, \tag{2.11}$$

$$w^{\text{IDF}}(q_i) = \ln \frac{N - n(q_i) + 0.5}{n(q_i) + 0.5}, \tag{2.12}$$

where q_i is the i^{th} term in the query, $n(q_i)$ is the number of documents containing q_i, N is the total number of documents, $tf(q_i)$ is the term frequency of word q_i in the document, dl is the document length, $avdl$ is the average document length in the collection, $w^{\text{BM25}}(q_i)$ is the score of the term q_i, w^{TF} and w^{IDF} are the term frequency and the inverse document frequency components of the BM25 score, respectively, and k_1 and b are configurable parameters. The model provides no guidance on how k_1 and b should be set, but experiments suggest that values between $0.5 < b < 0.8$ and $1.2 < k_1 < 2$ usually do well in many settings [37].

2.3 Question generation to improve training

The quantity of training data plays a big role in the quality of deep learning models [38]. Klein and Nabi [39] show that models trained by replacing some of the training data by synthetic data can reach performance close to models trained on the whole training set. This implies that automatic question generation is a viable strategy to perform data augmentation in a low-data regime QA. Klein and Nabi use a GPT-2 and a BERT models to automatically generate questions from SQuAD. They propose to improve GPT-2 ability to generate good questions by evaluating the generated questions with a BERT model for QA.

To generate question from documents, it is often necessary to first select answer phrases from documents [40, 41]. These are segments of the sentence that can serve as target for questions. It might be tempting to use entities, e.g., dates, people names, locations, as answer phrases; but even though entities correspond to over 50% of SQuAD v1.1 answers, for example, not all documents contain entities and not all entities are answers [40]. Subramanian et al. [40] use a pointer network to extract answer phrases from SQuAD. The pointer network is an RNN that learns to sequentially point to the start and end points of the answer segments. To generate the questions, they use a sequence-to-sequence framework with pointer-softmax decoder. This network allows to both generate new text on a generative fashion as well as copy tokens from the input to the output.

Alberti et al. [41] decompose question generation in three steps. First, a BERT model selects an answer phrase from the document. Then, conditioned on the selected answer phrase, a modified BERT or an original transformer model generates questions. The third step uses a BERT model trained for question answering and validates the questions by trying to answer it given the context. Questions that get the right answer are kept, the ones with the wrong answer are discarded. They show that using such synthetic data for training improves the results on both SQUAD v2.0 and NQ. In fact, using the synthetic data from the model trained in NQ improves also the results on SQuAD v2.0, but the reverse is not true.

Nema et al. [42] use a cascade approach to question generation. First, one network learns to create a draft question conditioned on a document and an answer. Then, a second network takes the question draft and refines it, generating the final question. This generation can be guided using a reward that promotes specific goals, like fluency and answerability. The networks are implemented using RNN/LSTMs and attention mechanisms.

2.4 Domain adaptation in question answering systems

Domain adaptation is any scheme where models are trained on source domains, for which we have labeled datasets available, and then evaluated on target domains, for which there is no labeled data available. The Machine Reading for Question Answering (MRQA) 2019 Shared Task [10] was designed to measure the performance of machine reading models on domain adaptation. Eighteen datasets where organized on a single

format: six dedicated for training, six for development/validation and six for evaluation.

D-NET [43] achieved the best performance in MRQA 2019. It uses an ensemble of XL-NET [44] and ERNIE [45], both transformer-based architectures. They apply multi-task leaning to train the network simultaneously on question answering and Masked Language Modeling (MLM) on an additional set of documents. The authors report that model selection is the most important aspect to improve generalization. Also, ensembles from models pre-trained on different sets of documents improves generalization performance, as well as the auxiliary MLM task.

Talmor and Berant [46] also study the performance of machine reading models on domain adaptation. They measure the performance of models trained on one or more datasets and transferred to another domain, with an optional fine-tuning phase. Results show a gap between in-domain and out-of-domain performance in most datasets. The performance in the target dataset is also dependent on the training dataset used. That could be a problem, since there is no way to know a priori which training dataset to choose, but results also show that the union of different training datasets performs similarly to the best single training dataset on domain adaptation.

Shakeri et al. [47] address domain adaptation from a question generation perspective. They propose a new model for question generation, based on the language model BART, that learns to generate QA-pairs in two stages. First, a context is fed to the model to produce a question. Then, both context and question are fed to the same model to produce a generative answer. If the answer is found in the context and its answer likelihood, as scored by the model, is among the top-50% QA-pairs produced by the context, it is added to the generated dataset. They show that their question generation model trained on SQuAD can generate questions on other domains, like biomedical articles in BioASQ, that improve performance when used during fine-tuning.

One research question that can be drawn from the domain adaptation by question generation technique presented above is how much of the performance gains can be attributed to the language modeling abilities learned by the model for facing question-answer-context examples in the target domain. In fact, the inclusion of target domain documents for pre-training, a technique called Domain-Adaptive Pretraining (DAPT), has been shown to improve out-of-domain performance in other NLP tasks [48].

In this work, we investigate how language modeling in the target domain improves generalization for QA Systems. Although the inclusion of domain documents for pretraining in QA has been investigated in previous work [1], we suspect that language models

were heavily under-trained with passages in the target domain. Therefore, we make a distinction between DAPT and the work of Nishida et al. [1], where DAPT relies on extensive pretraining of the language model on the target domain. In this work, we investigate how DAPT performs for QA and how it ranks in relation to the question generation approach of Shakeri et al. [47].

3 PROPOSAL AND CASE STUDY

This chapter describes our proposed framework for designing closed domain QA systems; experiments, comparisons, and implementations are described in later chapters.

Section 3.1 provides a formal structure for the main problem we have to tackle. Section 3.2 discusses the methodology and the reasoning applied throughout this work. Section 3.3 presents a central part of our work: the application of DAPT for QA, later applied to train the Reader. Finally, Section 3.4 presents our proposed architecture for domain-specific QA systems in the low data regime, and Section 3.5 describes our case study.

3.1 Reducing the cost of building a QA system

Finding labeled data for QA in specific domains can be challenging. Apart from some domains with already consolidated datasets, like biomedicine [49], news [50] and cinema [51], most domains do not have abundant labeled data available. One possible abundant source of QA pairs is search logs [8]. For instance, in our case study, Petrobras had a search engine in place for internal usage, from where a search log was collected. However, engineers resorted mostly to simple keyword-based queries. This probably reflects the engineers awareness that the current system could not handle semantic meaning. Keyword-based queries are not suitable as training data for QA, since they do not form a valid sentence in natural language.

Another option is to create labeled data from crowd workers. This was the approach taken by SQuAD when creating a large scale dataset for machine comprehension from Wikipedia articles. This approach does not generalize well to every domain, though. As the domain gets more specialized, as in our case study in offshore engineering, the skill level required to create interesting questions grows and so does the cost of creating the dataset.

We wish to create a framework for domain-specific factoid question answering that

does not require high initial costs into building the system. Hence, we frame our problem within a domain adaptation setting. We must label only the necessary data for evaluation, and leverage general domain and other domain datasets along with domain adaptation techniques to create a QA system on our target domain.

3.2 Our overall strategy

Following the literature, we divided our framework into two modules with distinct responsibilities: Retriever and Reader.

The Retriever is responsible for selecting potentially relevant documents from the document pool. It has to search among a large number of documents in a short period of time. It must have a good recall, as the system cannot recover from a failure in delivering an answer bearing document. It is also desirable for the Retriever to have a good precision, as this makes the work of the Reader easier.

The Reader is responsible for outputting the answers given a question and the relevant documents provided by the Retriever. We opted for extractive question answering in our framework: displaying the extracted answer together with its context is more interpretable than displaying generated text from a neural network, and therefore inspires more trust on the system. In an extractive QA setting, the Reader re-ranks the documents returned by the Retriever and selects inside those documents the spans that contain the answers.

The Reader is less constrained in respect to time efficiency, since it has a small set of documents to process. It can spend more time reading at each document, so the ranking produced by the Reader can better capture the semantics of questions and passages than the ranking produced by the Retriever.

Figure 3 presents an overview of the information flow in our framework. The Retriever accepts, as inputs, texts from documents and questions, and outputs a small set of relevant documents for each question. The Reader takes relevant documents from the Retriever and questions from users and re-ranks documents and pin-points the answer in them.

We have studied, as described in later chapters, two approaches for the Retriever: statistical and neural. The statistical approach is based on BM25, the standard ranking function employed in two of the main text search software available on the market, Solr and ElasticSearch. The neural approach is based on DPR, a state-of-the-art neural IR system based on BERT.

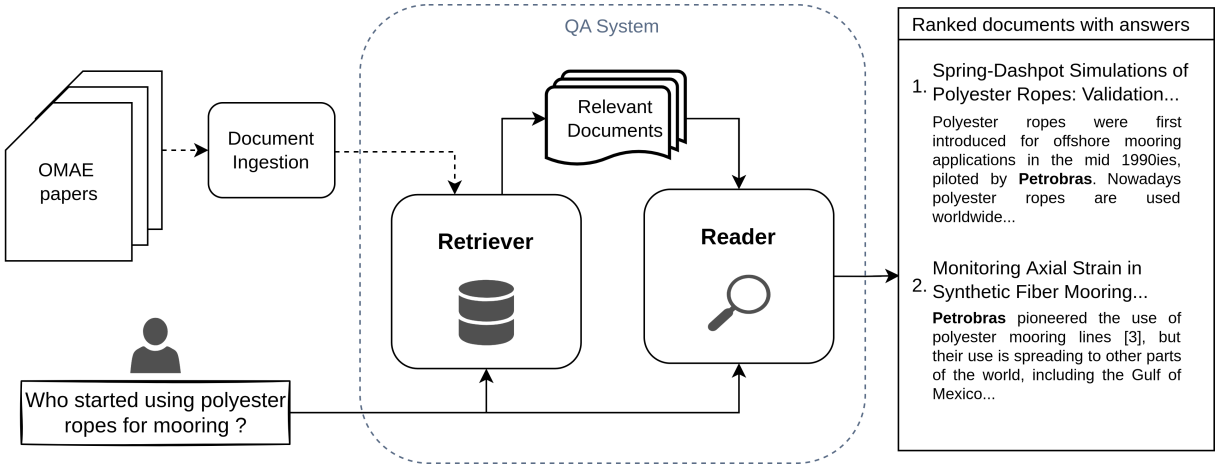


Figure 3: High-level overview of our framework. Dashed arrows represent operations prior to the system availability for users. Regular arrows represents the flow of information on the system when users issue questions.

BM25 uses the bag-of-words model for searching. By providing tolerance about small variations in words, text normalization can improve BM25 performance. We analyzed the effects of lowercasing, stemming, lemmatization, stopword removal and interrogative word (wh-word) removal. The bag-of-words model does not consider the position of words in a sentence. To mitigate this problem, we investigated the use of 2-grams and 3-grams while indexing.

The neural approach, on the other hand, searches for the semantic meaning of the sentences, and benefits from a well formed sentence. It can deal with variations of words and considers word positions in sentences and how each word relates to the others. Therefore, for the neural approach, we normalized text with lowercasing as the model has seen during training. We experiment with the trained model released by the original authors of DPR.

As discussed later, we settled for BM25 in the Retriever as it displayed better performance.

For the Reader, we assumed a neural approach based on DPR Reader and applied domain-adaptive pretraining for QA, which we describe in detail in the next section.

3.3 Domain-adaptive pretraining for QA

Most of the success of recent transformer-based neural networks can be attributed to powerful language representations built during pretraining. The model learns, during pretraining, relationships between words such that the model can, during finetuning,

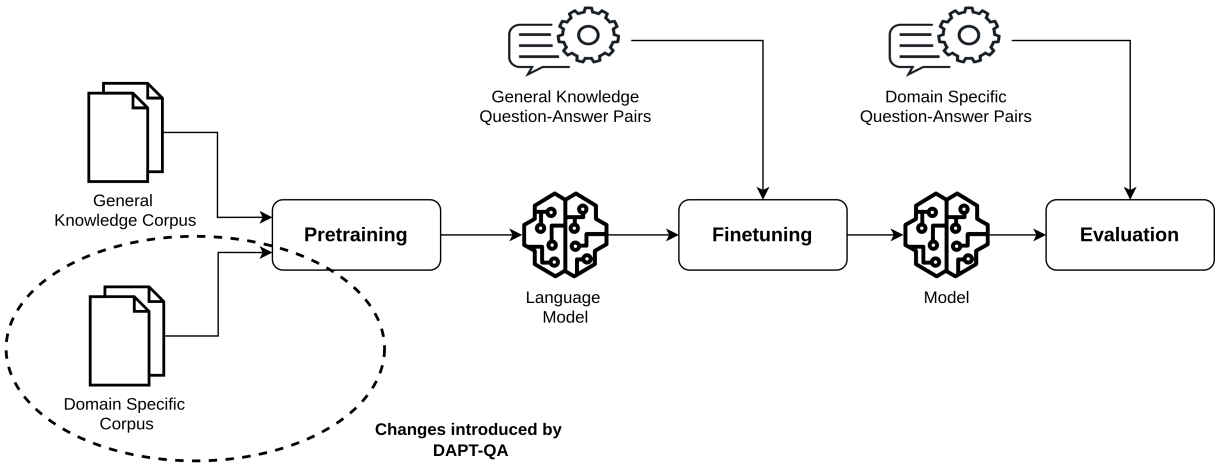


Figure 4: Domain-Adaptive Pretraining for QA.

better generalize from fewer examples.

Given that during pretraining the model learns relationships between words in the training corpus, we propose here the application of DAPT for QA (DAPT-QA). DAPT-QA consists of including, during pretraining, corpora on both source and target domains. By creating a joint representation for both domains, the model can better generalize to examples on the target domain even when presented, during finetuning, only with examples on the source domain. Figure 4 illustrates the changes introduced by DAPT-QA.

Without domain adaptation, the model is usually trained in two steps: first pretrained with a general knowledge corpus, then finetuned with the general knowledge QA pairs. This same model is used to answer questions in a specific domain.

In DAPT-QA, the model is presented with domain specific documents during pretraining, so as to learn a common semantic space between general knowledge and the domain at hand. In contrast with Gururangan et al. [48], we do not constraint the target domain sentences to a second pretraining phase. Models pretrained in two phases, as well as models trained in just one phase with documents involving the target domain, are treated equally.

It is well known in the literature that increasing the diversity of pretraining corpora favors the quality of pretrained language models [30]. It is also known that including documents related to the relevant domain improves performance when finetuning with in-domain data (i.e., not in a domain adaptation setting) [9]. DAPT-QA emphasizes the relevance of including target domain corpora when training a model in domain adaptation for QA.

Nishida et al. [1] have explored the use of target domain texts for language modeling to improve performance in domain adaptation. Shakeri et al. [47] have shown that their Question Generation approach can outperform the approach by Nishida et al. Our work differs from the work of Nishida et al. [1] as we invest in more deep pretraining with documents in the target domain. We show later that, by investing more effort in the unsupervised language model training, DAPT-QA outperforms the Question Generation approach while displaying a simpler training process.

3.4 The Low Data QA Framework

To summarize, as a result of our experiments we propose the following Low Data QA Framework. For the Retriever module, we propose an integration of BM25 with a custom text and query pipeline involving lowercasing, stemming, 3-grams and wh-words removal. In a low data setting, BM25 has the advantage of being an unsupervised retrieval model, and, therefore, it does not need training. The linguistic gap problem is a major concern in applications of BM25, since slightly variations of words can prevent BM25 from scoring passages correctly. We address that with selected text preprocessing and n-grams techniques.

For the Reader module we propose an association of the DPR Reader architecture and the DAPT-QA technique for training. DPR Reader training works by finetuning a transformer-based model for selecting the correct answer span among a set of passages and producing a rank. We propose pretraining the base language model on both the source and target domains, instead of using a default language model trained on general knowledge domain.

3.5 Case study: QA in offshore engineering

We ran a case study on the offshore engineering domain. This case study is another contribution of this work and functions both as motivation and as an experimental laboratory for our framework.

To develop and test objectively an IR system based on natural language questions and answers, we must have a set of relevant question/answer pairs that is related to the domain of interest. Several public repositories are available with question/answer pairs for open domain question answering, e.g., SQuAD [7, 29], Natural Questions [8] and WikiQA [52].

That is, these repositories contain general questions about a variety of subjects. Due to this, these repositories are not well suited for a closed domain question answering system; that is, a system that focuses on a specific domain such as offshore engineering. For this reason, we decided to build the Offshore QA Dataset, a dataset with question-answer pairs that we expect to see in production. This allows us to guide the development, track advances, and benchmark performance.

We followed a set of steps to perform our case study, namely:

- Creating the Offshore QA Dataset for evaluation of our framework;
- Implementing an IR engine to retrieve candidate documents from a document pool in offshore engineering;
- Implementing a Reader module to extract answers from candidate documents in offshore engineering;
- Validating the system using the Offshore QA Dataset.

The execution of each of these steps is covered in the remainder of this document.

4 VALIDATION OF DOMAIN-ADAPTIVE PRETRAINING FOR QA

In this chapter, we describe experiments to validate our approach for domain adaptation for QA (that is, DAPT-QA). Even though this domain adaptation strategy is only applied to the offshore engineering domain in Chapter 7, the application of domain-adaptive pretraining for QA is conceptually one of our main contributions, so we present its empirical validation here. We performed our validation experiments on the RC task, a NLP task close to QA but simpler and with more established literature and benchmarks.

4.1 Experiments

We use three datasets of different domains to validate DAPT-QA: SQuAD, on the general knowledge domain, NewsQA, on the news articles domain, and BioASQ, on the biomedical domain.

Pretraining of language models is very expensive due to the computation power required for the task. For this reason, we leverage three existing pretrained models trained on different sets of domains for our experiments: BERT-Base[1], RoBERTa-Base[2], and BioBERT-v1.1 [3] [9], a BERT model specialized on the biomedical domain. Our goal is to validate our hypothesis that, in a domain adaptation setting, RC models trained from language models that know both source and target domains out-perform models trained from LMs that do not know the target domain.

Table 1 presents the corpora on which each model was pretrained by the original authors. The English Wikipedia is an online encyclopedia created and edited by volunteers. The BookCorpus is a corpora of novel books. CC-News is a corpora containing 63 million news articles. OpenWebText contains web-documents extracted from urls on Reddit with at least three upvotes. Stories contains texts to target common sense reasoning; and

[1] huggingface.co/bert-base-uncased
[2] huggingface.co/roberta-base
[3] huggingface.co/dmis-lab/biobert-base-cased-v1.1

Table 1: Corpus used for pretraining on each pretrained model.

Corpus	BERT	RoBERTa	BioBERT
English Wikipedia	X	X	X
BookCorpus	X	X	X
CC-News		X	
OpenWebText		X	
Stories		X	
Pubmed			X

Source: Refs. [9], [28] and [30]

Pubmed is a corpus of abstracts from scientific articles on biomedicine.

In order to evaluate how the different choices of pretraining corpora affects domain adaptation, we finetune each model in one of the two large datasets, SQuAD and NewsQA, and evaluate in all three datasets in order to measure in domain and out-of-domain performances. We train for 4 epochs, with batch size 24, learning rate 3×10^{-5}, AdamW optimizer [53] and gradual layer unfreezing in order to stabilize training and prevent over-fitting.

We hypothesize that a base LM pretrained on both source and target domains can better correlate the domains and, therefore, produce better results in the domain adaptation setting. Hence, we expect RoBERTa to yield the best performing model in NewsQA, as it has seem documents from news articles during pretraining, and BioBERT to yield the best model on BioASQ, as it has seem biomedical documents during pretraining.

We report Exact Match(EM) and F1-score metrics as defined by Rajpurkar et al. [7]. EM measures the percentage of predictions that match any one of the ground truth answers exactly. F1-score measures the average overlap between the prediction and ground truth answer. It treats prediction and ground truths as bag of tokens, and compute their F1. The maximum F1 over all ground truths for each question is computed, and its average among all questions produces the F1-score.

All three base language models mentioned above were trained with massive amounts of data. Moreover, increasing the volume and diversity of data seems to be an important player regarding the quality of a language model [30].

A major concern regarding our approach for domain adaptation can, therefore, be the volume of text available for pretraining. To address this issue, we further pretrain

BERT-base uncased on the biomedical domain using different proportions of the PubMed corpus: one sixteenth, one quarter, one half and the totality of the corpus. On each trial, we train the model for 200k steps, which corresponds to about one epoch on the complete PubMed corpus. Our goal is to measure the performance of the model in a more text-data restricted scenario in comparison with the text-data abundant scenario.

We use an adaptation from the original BERT script to generate the training examples and use duplication factor equals to four for the run with one sixteenth of PubMed and one for the others. Unless otherwise noticed, we use the same hyperparameters as BioBERT v1.0.

4.2 Results

Table 2 presents the Exact Match (EM) and f1-scores produced by BERT, RoBERTa and BioBERT finetuned on SQuAD or NewsQA and evaluted on all datasets. RoBERTa has the best in-domain performance in both SQuAD and NewsQA, but, for domain adaptation on the biomedical domain, BioBERT outperform the other two models in both settings. RoBERTa, the only model to include the news domain, also is the best model in adapting from SQuAD to NewsQA.

When trained with SQuAD, BERT and BioBERT have similar performance both in-domain and on NewsQA, a domain not seen during training for both, but, when transferred to the biomedical domain, BioBERT outperformed BERT by a large margin (10 pts EM).

These results provide empirical evidence to our hypothesis that models trained on both source domain and target domain will be able to generalize better to the target domain. Table 3 compares our DAPT-QA approach to the literature. Our approach establishes new state-of-the-art scores for domain adaptation from SQuAD to both NewsQA and BioASQ.

Regarding the volume of data required for DAPT-QA, Figure 5 shows our resulting learning curve on data for our trials. The metrics are reported for each model finetuned on SQuAD and evaluated on BioASQ. Also, to observe the influence of training time, we evaluate each model at 100k and 200k training steps.

No clear trend can be observed regarding the availability of data for training. Even using only one sixteenth of the training corpus, which corresponds to 243M words, the performance does not degrade. This observation reinforces the feasibility of our approach

Table 2: In-domain and out-of-domain performance when trained on source and evaluated on target datasets. Columns represent the dataset and base language model used for training. Exact Match and F1 (in parenthesis) are reported. Best values for domain adaptation for each target dataset on each source dataset are highlighted in bold.

	SQuAD			NewsQA		
Target	BERT	RoBERTa	BioBERT	BERT	RoBERTa	BioBERT
SQuAD	-	-	-	63 (78)	**70** **(84)**	65 (79)
NewsQA	38 (55)	**45** **(63)**	37 (53)	-	-	-
BioASQ	41 (55)	48 (61)	**51** **(63)**	33 (52)	34 (55)	**36** **(60)**
In-domain	80 (88)	84 (91)	81 (88)	52 (68)	58 (73)	51 (66)

Table 3: Comparison of DAPT-QA with the state-of-the-art when performing domain adaptation from SQuAD. DAPT-QA achieve better performance on both datasets. Note that Nishida et al. [1] does not use the MRQA version of NewsQA, so it is not directly comparable with the other results.

	NewsQA		BioASQ	
	EM	F1	EM	F1
Nishida et al. [1]	-	-	45.4	57.8
Shakeri et al. [47]	45.04	60.79	48.40	58.33
DAPT-QA (ours)	**45.06**	**62.74**	**50.73**	**63.48**

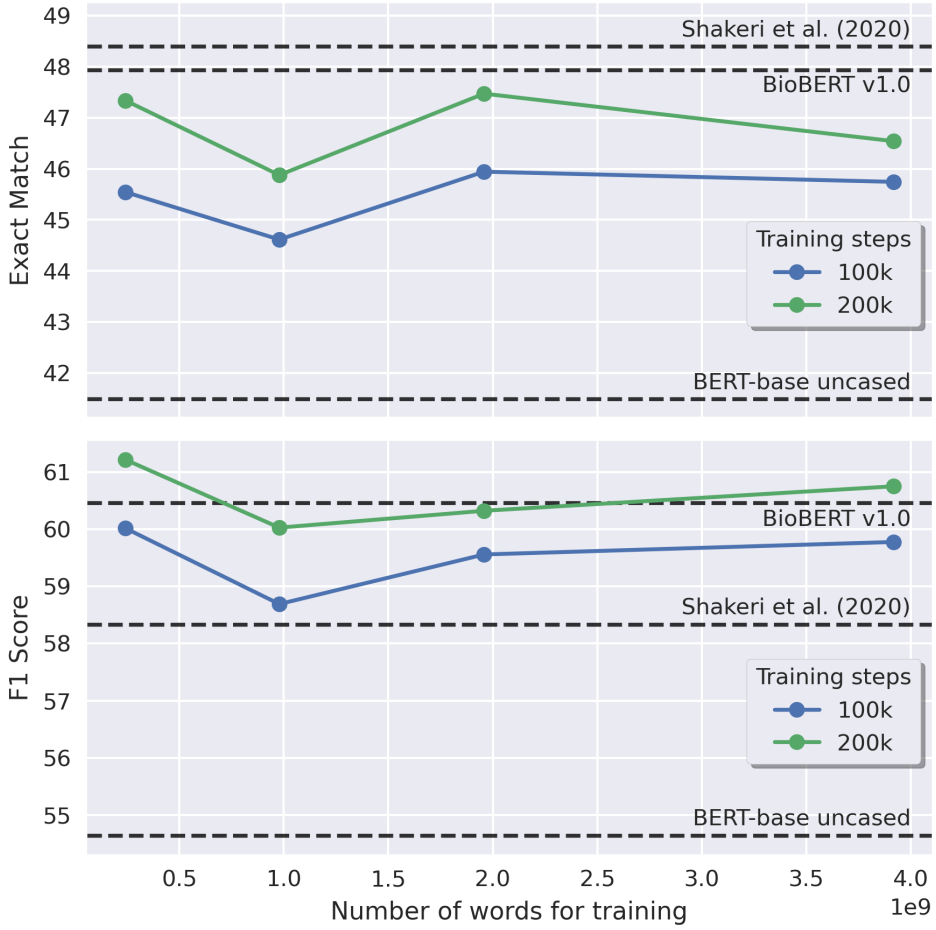


Figure 5: Learning curve for the DAPT-QA approach from SQuAD to BioASQ using data from PubMed to further pretrain BERT-base uncased. Each point in the plot is the result of the sequence of first further pretraining BERT-base on a subset of #N words from Pubmed and then using this LM as base for RC finetuning with SQuAD and evaluating on BioASQ. As reference, the metrics obtained from other models in the same domain adaptation settings are represented in dashed lines.

for real life scenarios, where domain corpus might not reach billions of words.

5 THE OFFSHORE QUESTION ANSWERING DATASET

The evaluation of any Question Answering technique or system requires a dataset containing questions and answering for testing. In this work such a dataset was built and used in many experiments related to the offshore engineering domain. In this chapter we describe the dataset; the contents here are not directly related to the main contributions but the dataset is relevant in several later discussions.

Two approaches for building a question answering dataset were studied. The first, based on the SQuAD methodology [7], produces questions about specific paragraphs of documents. The second, based on the Natural Questions methodology [8], collects questions from search logs and finds answers among the results returned by a given information retrieval system.

Questions generated with the Natural Questions method are known to be better at representing the nature of QA, where users have incomplete information when issuing a query [33]. Hence we started with this method. We first analyzed logs gathered from the existing system at Petrobras. They are mostly populated with keyword-based searches, so they did not help in building a dataset representative of natural language questions. We believe that the keyword-based behavior is prevalent due to the fact that users have significant knowledge of the current system, that operates with a very limited semantic understanding of any query.

We thus decided to build a dataset following the SQuAD methodology. That is, given a text paragraph, we formulated questions that have as response a segment of that paragraph. The questions were formulated using articles from the International Conference on Ocean, Offshore & Arctic Engineering (OMAE), keeping in mind the kind of information a person not looking at that exact article would be looking for. Therefore we avoided asking questions about information that would be too specific about a particular paper. Two annotators were designated to build this dataset. Due to time constraints, we kept this dataset to a very limited size, with 100 question-answer pairs. Table 4 shows

Table 4: Samples from our manually constructed question answering dataset based on OMAE papers. Questions styled according to their corresponding answers in the passage.

P1	Polyester ropes were first introduced for offshore mooring applications **in the mid 1990ies**, piloted by ***Petrobras***. Nowadays polyester ropes are used worldwide, particularly for deep-water applications where catenary systems become heavy and inefficient. (...)
Q1	**When did polyester ropes start being used for mooring?**
Q2	***Who was the first company to start using polyester ropes for mooring?***
P2	A jack-up rig has to be designed for extreme storm conditions in its elevated mode during operations. Guidelines of **ISO 19905-1 and SNAME TR-5-5A** for site specific assessment of jack-up rigs explain in detail such analysis and assessment requirements. (...)
Q3	**Which norms regulate the construction of jack-up rigs?**

samples of the question-answer pairs obtained for this dataset.

From our experiments, we noticed that the QA system can return multiple answers that are correct but are not the golden answer in the dataset. For instance, for the question:

> *When have polyester ropes started being used for mooring?*

the system finds the answer (in bold):

> *The world-first polyester synthetic rope TMS was installed in **1997** by Petrobras Brazil*

which answers the question but is different from the golden answer "*mid 1990ies*". Another situation is when answers are found in different variations of the text. For instance, for the question:

> *What is the temperature difference required for ocean thermal energy generation?*

with golden answer "*20 C*", the system finds the answers:

- 20 degree centigrade,
- at least 20oc,

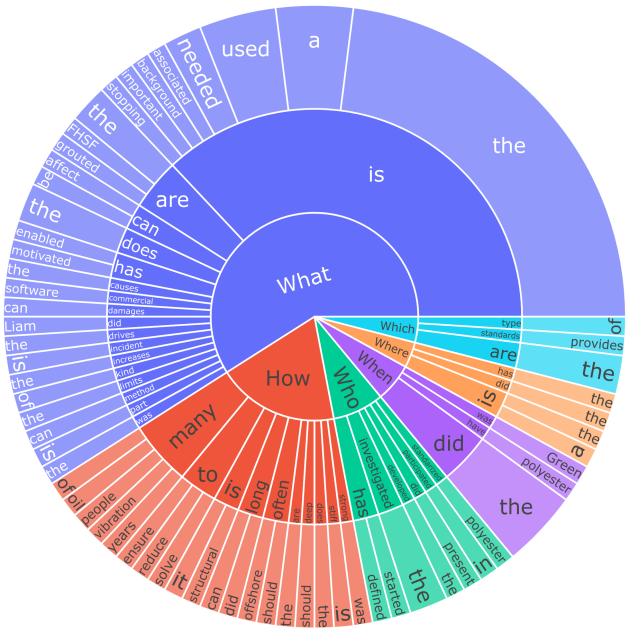


Figure 6: Distribution of the first three words in the Offshore Question Answering Dataset.

- $>$ 20 degrees celsius.

These answers all represent the same information. It is easy for a human to verify they are correct, but it is hard to judge them correctly based on automatic evaluation metrics, like exact match or F1-scores.

In our dataset, it is important that evaluation metrics can be calculated without human supervision so we can test and compare different approaches to the QA system of interest. Taking into account the answer diversity stated above, we take a compromise between efficiency and reliability. The most reliable measurement is obtained when a human verify the answers for each question. The most efficient measurement is obtained when answers are judged automatically. We run manual evaluations for top-5 answers in a few experiments and save the alternative answers found as part of our answer set in the dataset. This extended answer set enable later automatic evaluations to better resemble human evaluation on answer diversity. We refer to the dataset with only the original golden answers as original dataset, and to the one with additional found answers as extended dataset. Unless otherwise noticed, we use the extended dataset in our experiments.

Table 4 shows samples from the dataset we built, the Offshore QA Dataset. Figure 6 show the distribution of the first three words in questions. It is evident the presence of the

Table 5: Metrics for random retrieval of 100 paragraphs using 1,000 Monte Carlo trials.

Answer set	MAP	MRR
Original	0.01	0.02
Extended	0.02	0.03

words *what, how, who, where and which* in the beginning of questions, which emphasizes the factoid nature of these questions. Figure 7 shows the distribution of the answer exact occurrences on paragraphs in the document collection, both for the original golden answer set as for the extended golden answer sets. This can help understand how prone our automatic evaluation system is to score passages retrieved as relevant just by chance. We also calculate MAP and MRR for a random retriever for both sets of answers (original and extended) and report on Table 5. The results are very close to zero, indicating that the answers are rare in the corpus and, therefore, the automatic evaluation is unlikely to provide good scores just by chance.

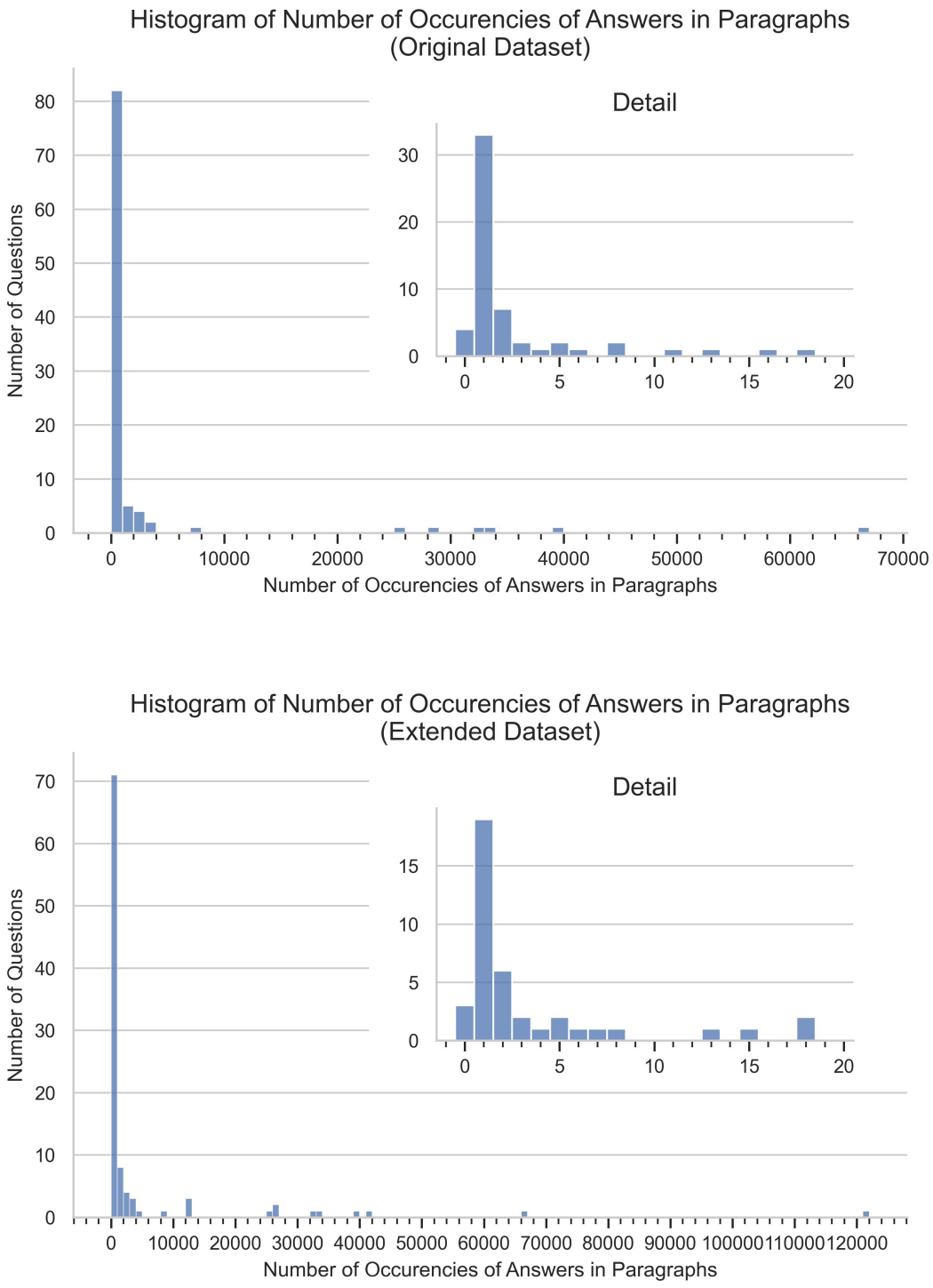


Figure 7: The number of occurrences of answers in paragraphs for the original (top) and extended (bottom) Offshore QA Dataset; there is a total of 446,838 paragraphs.

6 THE RETRIEVER: BM25 VS. DPR

In this chapter we describe the experiments we performed so as to reach our proposed Retriever strategy. In short, we compared a statistical (BM25) and a neural (DPR) strategy in the context of offshore engineering so as to grasp the behavior of these techniques in a specific domain question answering setting. Our conclusion was that BM25, with properly tuned resources, is the best retrieving strategy. Even though our conclusion is obviously restricted to a particular domain, it seems possible to generalize some of our insights to other domains and to adopt them as a sensible practical path.

6.1 Experiments

We use 12,890 papers from OMAE 1998-2019 conferences as documents for retrieval. We extract text from pdf files as reported by Gomi et al. [54]. Queries used on the experiments are questions in the Offshore QA Dataset, and documents are judged relevant when they contain the exact string of the answer.

We used Mean Average Precision (MAP) as a metric. Precision is the proportion of relevant results for a query. Average Precision (AP) is the average of the precision considering only the documents ranked up to each relevant document. Mean Average precision is the average AP among queries. Let Q be the total number of questions, m_i the number of correct results in the i^{th} question, and p_j^i the ranking position of j^{th} correct result of the i^{th} question. Then MAP is:

$$\text{MAP} = \frac{1}{Q}\sum_{i=1}^{Q}\left(\frac{1}{m_i}\sum_{j=1}^{m_i}.\frac{j}{p_j^i}\right) \tag{6.1}$$

We also report Mean Reciprocal Ranking (MRR) and Hit@N metrics. Reciprocal Ranking (RR) is one over the ranking of the first relevant results, or zero, if no relevant result is found. MRR is the average RR among queries. Hit@N is the proportion of queries that have at least one relevant document up to ranking position N. MRR is calculated as

follows:

$$\text{MRR} = \frac{1}{Q}\sum_{i=1}^{Q}\frac{1}{p_1^i}. \tag{6.2}$$

Both MAP, MRR and Hit@N have the worst value at zero and the best value at one. While all of these metrics represents the quality of the ranking, MRR only cares about the ranking of the first relevant documents. Hit@N only measures the presence of the results, demanding the use of several N values in order to get a feeling of how the ranking behaves. MAP, on the other hand, captures both the ranking and the position of every relevant result in a single number. For this reason, we choose MAP as the main metric for our evaluations.

6.1.1 DPR vs BM25

Our first experiment aims at evaluating the performance of state-of-the-art neural and statistical retrievers. For BM25, we use Elasticsearch.[1] Elasticsearch is based on Apache Lucene under the hood, which is responsible for indexing and retrieving documents. For DPR, we use the pre-trained models available through Hugginface's Transformers library [55].

For this experiment, we split documents in paragraphs and paragraphs in passages of 100 words, following the procedure used in DPR. DPR is bound by 512 BERT tokens while BM25 does not have a limit in document size. However, in order to establish a fair comparison between DPR and BM25, we index the same passages in both retrieval engines. Allowing BM25 to retrieve whole documents would increase drastically the probability of the answer being present just by chance, as it would be retrieving much more text, and indexing by document and retrieving just to match the number of words retrieved by DPR would limit the flexibility of BM25 and therefore give an advantage for DPR.

We use default index parameters for Elasticsearch, which only lowercases text before indexing. For DPR, we experiment using document title and passage as well as an empty title and passage. We report the best result.

6.1.2 BM25: documents units

For this experiment we aim to evaluate how different document units (document, paragraph, 100-word passage) affects the performance of the Retriever. Larger units of

[1] https://www.elastic.co/elasticsearch/

text tend to perform better on the metrics, for the reasons cited on section 6.1.1. So, we also report the average total words (Words@N) on retrieved documents at different numbers of documents.

We index whole documents, documents split in paragraphs and paragraphs split on 100 word passages on Elasticsearch and compare the performance of retrieval according to MAP, MRR, Hit@N and Words@N.

6.1.3 BM25: text and question pre-processing

When indexing with DPR, there is not much flexibility about how to input documents into the Retriever. DPR accepts a passage and the document title and builds the index. It automatically deals with word variations and lexical mismatch between query and passages. BM25, on the other hand, requires much more tuning regarding text normalization, dealing with the lexical gap problem and custom engineered texts pipelines in order to approach the query from the relevant passages style. Therefore, for BM25, we study how different text normalization, indexing options and question pre-processing affects performance. For text normalization we evaluate how lowercasing and stemming affects the system. We also investigate on stopword removal and n-grams. For question preprocessing, we study wh-word removal.

6.2 Results

Table 6 shows the results for the comparison between DPR and BM25. BM25 performs significantly better than DPR in our domain. This may be due to the differences between documents used during DPR training and documents from our case study. Machine learning systems tend to degrade when applied in settings distinct from training. Articles from OMAE are different from Wikipedia articles, and DPR seems not to be able to generalize well to this new set of documents. BM25 does not face such problems as it is a purely statistical retriever. On the other hand, the construction of the QA dataset may favor BM25 retrieval [33]. Questions created with knowledge of the answer passage tend to have a higher overlap of words with the passages than questions created otherwise. This word co-occurrence favors BM25, which relies on this information for retrieval. In absence of further evidence that the performance gap between DPR and BM25 is due to the limitations in dataset construction, we conclude that BM25 is a better approach, although this comparison can be reviewed in the future in case a search-log QA dataset

Table 6: DPR and BM25 performance on the Offshore QA Dataset. BM25 does better than DPR according to all metrics.

	Metrics					
			Hit at			
Sistema	MAP	MRR	1	5	10	100
DPR	0.15	0.27	0.16	0.41	0.45	0.72
BM25	**0.31**	**0.48**	**0.37**	**0.60**	**0.68**	**0.87**

Table 7: BM25 performance with different document units. For all experiments, 100 units are retrieved for each question. Note that while metrics using the unit *Document* provides better results, it does so because of the much larger volume of text retrieved when indexing whole documents.

	Metrics									
			Hit at				Words at			
Doc. Unit	MAP	MRR	1	5	10	100	1	5	10	100
Document	0.41	0.60	0.50	0.71	0.79	0.94	3931	18725	37289	398012
Paragraph	0.31	0.48	0.36	0.61	0.66	0.88	112	506	1003	9820
Passages	0.32	0.50	0.39	0.65	0.71	0.88	89	437	865	8583

is constructed.

Table 7 shows the retrieval metrics for indexing different document units in BM25. The MAP for the documents is much larger that the other units, but this comes at a cost of retrieving a much larger volume of text from the document pool. The role of the Retriever is to select relevant information. An excess of text can hurt performance downstream, on the Reader module, and also slow the response, as the Reader module takes some time to process each document. For the same volume of text, retrieving with paragraphs or passages can recover more information, as evidenced by the passage retrieving that recovers an average of 8,583 words as 100 passages and brings answers to 88% of questions while document retrieving recovers an average of 18,725 words at 5 documents and brings answers to only 71% of questions. For that reason, we consider paragraph and passage indexing to be better than document indexing, as they recover more relevant information given a limited amount of text.

Using paragraphs or passages as document units seems to be very close in effectiveness for retrieval, but passages perform slightly better. Therefore, we select passages as the document unit for retrieval.

Table 8 shows the results of the experiments with different pre-processing parameters for indexing with BM25. In these experiments, we index documents split in passages. Figure 8 shows the improvements of each parameter for retrieval. Each improvement is

calculated based on the results with the parameter set in relation to the same configuration but with the parameter unset. So, for example, from the first and second rows in Table 8, we derived a point that wh-words removal improves MRR by 0.01. From Figure 8, it can be noted that lowercasing always improves performance. Stemming also helps most of the time. These can be expected, since text normalization assists into solving the more shallow lexical gap problems. Wh-words removal is also always helpful. Wh-words are highly linked to queries, and, even if they are found in the documents, they are most likely being used with another meaning, so including these words in queries only mislead the Retriever. 2-grams also helps most of the time and it is also noticeable that using 3-grams does not provide further benefits to 2-grams. Stopword removal consistently hurt performance. This can be due to some of the default english stopwords used by Elasticsearch being useful for retrieval.

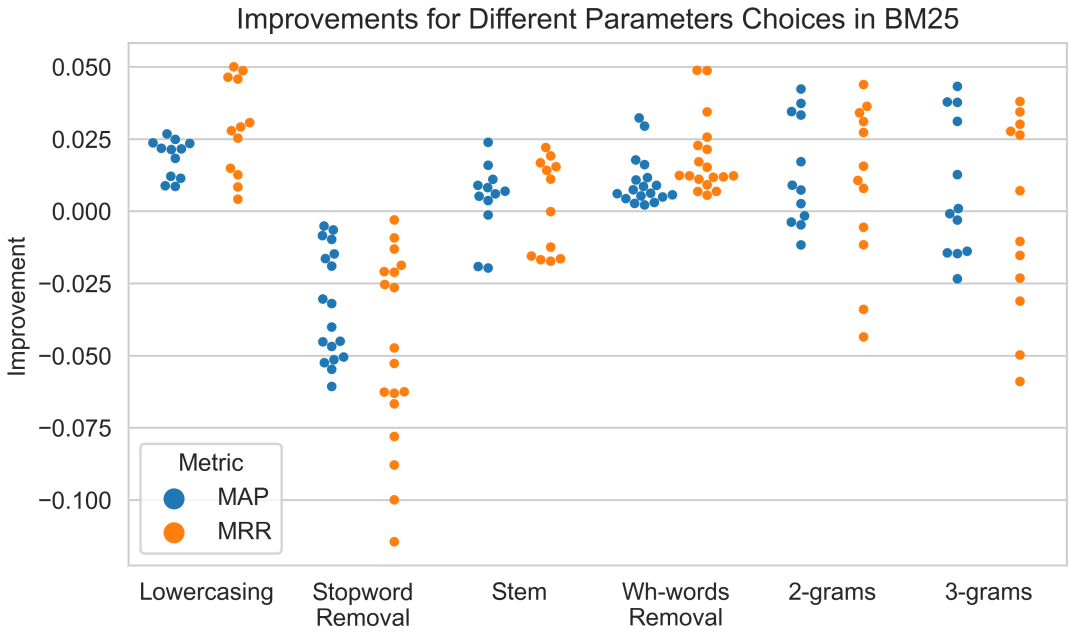


Figure 8: Improvements with respect to different parameter choices in BM25. Each point represents the retrieval improvement produced by setting the parameter in relation to the same configuration but with the parameter unset. Values are calculated from Table 8.

Table 8: Experiments with text and query pre-processing with BM25. Best performance is achieved when using lowercasing, stemming, n-grams and wh-words removal. Figure 8 gives a better idea of the particular contributions of each of those pre-processing steps in the overall result.

Parameters					Metrics					
Document Processing				Question Processing			Hit at			
Lower-casing	Stopword Removal	Stem	N-grams	Wh-words Removal	MAP	MRR	1	5	10	100
					0.29	0.45	0.33	0.58	0.64	0.84
				X	0.29	0.46	0.33	0.59	0.64	0.84
			2		0.32	0.48	0.37	0.60	0.65	0.85
			2	X	0.34	0.49	0.39	0.60	0.66	0.86
			3		0.33	0.48	0.37	0.60	0.67	0.85
			3	X	0.34	0.49	0.39	0.60	0.68	0.85
	X				0.28	0.44	0.33	0.57	0.63	0.84
	X			X	0.29	0.45	0.34	0.58	0.63	0.84
	X		2		0.29	0.46	0.38	0.56	0.65	0.83
	X		2	X	0.30	0.46	0.39	0.56	0.65	0.83
	X		3		0.28	0.43	0.34	0.55	0.61	0.82
	X		3	X	0.29	0.44	0.35	0.55	0.61	0.82
X					0.32	0.48	0.37	0.62	0.70	0.87
X				X	0.32	0.50	0.40	0.63	0.70	0.87
X			2		0.35	0.52	0.43	0.62	0.69	0.88
X			2	X	**0.36**	0.54	**0.45**	0.65	0.69	0.88
X			3		0.35	0.51	0.40	0.63	0.69	0.88
X			3	X	**0.36**	0.54	**0.45**	0.65	0.69	0.88
X		X			0.33	0.50	0.40	0.61	0.69	0.87
X		X		X	0.34	0.52	0.43	0.63	0.70	0.89
X		X	2		0.33	0.51	0.37	0.65	0.71	0.89
X		X	2	X	**0.36**	**0.55**	**0.45**	0.67	**0.73**	**0.90**
X		X	3		0.33	0.51	0.37	0.67	0.71	0.88
X		X	3	X	**0.36**	**0.55**	**0.45**	**0.69**	**0.73**	**0.90**
X	X				0.31	0.47	0.36	0.61	0.67	0.87
X	X			X	0.31	0.48	0.38	0.61	0.68	0.87
X	X		2		0.30	0.46	0.38	0.57	0.64	0.86
X	X		2	X	0.31	0.47	0.39	0.57	0.66	0.86
X	X		3		0.29	0.44	0.35	0.55	0.62	0.86
X	X		3	X	0.30	0.45	0.36	0.56	0.63	0.86
X	X	X			0.31	0.48	0.37	0.60	0.68	0.87
X	X	X		X	0.33	0.50	0.39	0.63	0.72	0.88
X	X	X	2		0.31	0.44	0.33	0.62	0.66	0.89
X	X	X	2	X	0.32	0.45	0.34	0.62	0.68	0.89
X	X	X	3		0.30	0.43	0.32	0.58	0.63	**0.90**
X	X	X	3	X	0.30	0.44	0.33	0.59	0.64	**0.90**

7 THE READER: OFFSHOREBERT AND OFFSHOREREADER

The Low Data QA Framework is composed of two modules: Retriever and Reader. In Chapter 6, we presented the experiments that led to the selection of the statistical BM25 for the Retriever module. In Section 3.2, we presented DAPT-QA and, in Chapter 4, we have shown experiments to validate this approach and shown that it achieves state-of-the-art performance in domain adaptation. In this chapter, we complete the Low Data QA Framework in the context of offshore engineering.

In the experiments described on Chapter 4, we applied DAPT for RC. As mentioned in Section 3.2, RC is a task close to QA. The difference between them is that, while in RC the model has only one document where to find the answer, in QA you have a collection of documents. In the extractive QA setting, the model must rank the documents and points the answers inside them, such that the most relevant documents are at the top, and the answer inside them are marked correctly.

The essence of DAPT-QA is in leveraging the underlying pretrained language models trained on both source and target domains to improve out-of-domain performance. Therefore, this principle can be applied to any model that uses the pretraining-finetuning approach of training transformers. In our case study, for the Reader module, we use DPR Reader. First, we further pretrain BERT on documents on the offshore engineering domain. We call the language model outcome of OffshoreBERT. We then use OffshoreBERT as base language model for DPR Reader training. In doing so, we close our question answering pipeline.

7.1 OffshoreBERT

We collected 3 corpora so as to pretrain OffshoreBERT:

OMAE: a corpus of full-text scientific papers from OMAE conference from 1998 to 2019 containing a total of 12,890 documents and 39 million words.

Table 9: NSP + MLM loss for language modelling with BERT and OffshoreBERT.

	Train	Dev	OffshoreBooks	OffshoreTechnical
BERT	5.0616	5.0919	5.5035	7.602
OffshoreBERT	**1.4714**	**1.5523**	**2.0835**	**2.1558**

ScienceDirectOffshore: a corpus of full-text scientific papers from 13 journals under the "Ocean Engineering" subdomain in Science Direct. Full-text for 43,080 documents are collected, totaling 199 million words.

ScopusOffshore: a corpus of articles abstracts from 191 publications under the "Ocean Engineering" subject area in Scopus. Abstracts for 257,790 entries are collected, from which 222,310 are not empty and 207,300 do not appear in the ScienceDirectOffshore corpus.

We access the quality of OffshoreBERT by using a 1% holdout dev set on the previous corpora and also two other corpora used only for the final evaluation:

OffshoreBooks: a corpus with content extracted from text books on offshore engineering totaling 741 thousand words.

OffshoreTechnical: a corpus of norms and technical documents on offshore engineering consisting of 61 thousand words.

We pretrained BERT starting from the checkpoint released by original BERT authors in two phases, following recommendations from the official repository[1]. On the first phase, we train using 128-token sequences and 768 batch size for 180k steps, 5e-5 learning rate with linear decay and 18k warm-up steps. On the second phase, we train using 512-token sequences, 192 batch size, therefore keeping the same number of tokens per batch, for 20k steps, 2e-5 learning rate with linear decay and 2k warm-up steps.

Table 9 show the comparison between the language modelling performance of BERT and OffshoreBERT. We report the joint NSP + MLM loss for the training and dev sets as well as the two evaluation corpora. Perplexity is a common metric used for measuring language model performances. However, perplexity is not fit for BERT since BERT is not an autoregressive model. We report on Table 10 the exponential of the MLM loss for the two evaluation corpora instead, which has the same dimension as the perplexity and can be also be interpreted as the inverse of the geometric mean of token probabilities in the sentence.

[1]https://github.com/google-research/bert#pre-training-tips-and-caveats . Accessed on Dec. 12, 2021, commit eedf5716ce1268e56f0a50264a88cafad334ac61

Table 10: Exponential of the MLM loss for language modelling with BERT and OffshoreBERT in the two evaluation corpora.

	OffshoreBooks	OffshoreTechnical
BERT	35.6193	32.2608
OffshoreBERT	**6.5972**	**5.1169**

Table 11: DPR Reader and OffshoreReader performance on the Offshore Question Answering Dataset. Metrics on the ranking produced by the Retriever are displayed for reference. P-values are obtained using paired t-tests on DPR Reader and OffshoreReader results.

			EM@N				F1@N			
	MAP	MRR	1	5	10	100	1	5	10	100
Retriever	0.36	0.55								
DPR Reader	0.41	0.56	**0.3**	**0.42**	**0.48**	**0.58**	**0.39**	0.55	0.6	0.73
OffshoreReader	**0.48**	**0.64**	0.26	**0.42**	**0.48**	0.54	**0.39**	**0.58**	**0.63**	**0.75**
p-value	0.016	0.054	0.348	1	1	0.374	0.905	0.450	0.414	0.437

OffshoreBERT shows a significant improvement for language modeling in the offshore engineering domain even when presented with documents with different formats from the ones seen during training, like on the OffshoreTechnical corpus. This shows that OffshoreBERT modelling is not tied to a specific document structure, but it is actually modelling the language in offshore engineering.

7.2 OffshoreReader

To create the Reader module for our case study, we train DPR Reader starting from OffshoreBERT as base language model. We train it for 18 epochs, choosing the best model based on the validation set. We use the same labeled data - Natural Questions dataset - and training regime as in the original DPR Reader. We call the resulting model the OffshoreReader.

Table 11 shows the comparison between DPR Reader and OffshoreReader on the Offshore Question Answering Dataset. Both models are fed with outputs from the Retriever module. OffshoreReader shows statistically significant improvement over the ranking produced by DPR Reader. The quality of the pin pointed answers within the documents seems to remain the same. Nevertheless, the increase in ranking performance is an advantage of OffshoreReader and it is directly associated with the better comprehension of the domain enabled by OffshoreBERT.

8 CONCLUSION

In this work we proposed a framework for closed domain QA systems design in the low-data regime: the Low Data QA Framework. The framework is composed of two main modules: (1) the Retriever, an IR system, responsible for finding relevant documents in a large document pool, and (2) the Reader, responsible for ranking the documents provided by the Retriever and pinpointing answers in them. We studied two approaches for the Retriever, one statistical and one neural, and proposed a Retriever composed of a BM25 index along with a custom text and query processing pipeline according to results observed in our experiments. For the Reader, we approached the problem from a domain adaptation perspective in order to avoid the costs associated with building an specialized training set for the target domain. We applied DAPT for QA, in which domain adaptation is performed by pretraining the underlying language model of the Reader on both documents on the source and target domains. We showed that this approach outperforms recent results on the literature, and that its performance does not degrade even with modestly sized corpora - in the order of 200 million words.

As case study, we applied our framework on the offshore enginnering domain. We built a small QA dataset specialized on the offshore domain, the Offshore QA Dataset, used for benchmarking. We collected 5 corpora in offshore engineering and pretrained a specialized language model: OffshoreBERT. OffshoreBERT was used as base language model for finetuning the Reader, in a DAPT-QA fashion. OffshoreBERT can also be employed in other tasks of NLP regarding the offshore engineering domain. We leave this exploration to future work.

Despite the recent progress in neural information retrieval, its models are sensitive to document type and degrade as input documents depart from documents used for training. This is a general difficulty with approaches based on machine learning, and its impact can be seen when comparing the performance of BM25 and of DPR in the Offshore QA Dataset. Even though DPR is able to retrieve relevant documents for most questions, BM25 displays much higher performance. Indexing with BM25 yields more flexibility

with respect to the indexed document units; however, it also requires more hand-designed processes for text normalization and for matching queries and documents. We showed that indexing 100-word passages performs similarly to indexing paragraphs and better than indexing whole documents (given the large amount of "noise" retrieved in the latter case). We also run a detailed analysis on how lowercasing, stopwords removal, stemming, n-grams and wh-word removal in queries affects our retrieval system. Lowercasing and wh-words removal consistently improve performance. Stemming and n-grams seem to help in most cases, and removing stopwords consistently hurts performance. The best combination we found consists of applying lowercasing, stemming, 3-grams and wh-words removal.

The interaction between question and documents in retrieval is very shallow. For performance reasons, they are only allowed to interact through a vector similarity function: euclidean distance, dot product, cosine similarity, etc. The real power of Transformer-based architectures lies in allowing the words to interact together in multiple layers of attention . This level of interaction demands execution time that is not available for the Retriever, but is available for the Reader module. Therefore, it is in the Reader module that the benefits of applying deep learning are most promising.

We showed on our case study that the ranking produced by the Reader improves MAP in 0.12 and MRR in 0.11, which corresponds to 33% and 16%, respectively, in relation to the ranking by the Retriever. Regarding the pin-point of answers within documents, considering the top 10 passages for each question, the Reader was capable of correctly marking the answers for 48% of the questions. This combination of better ranking and a objective tag of the answers within document leads to more productivity and overall better search experience for the user.

The DAPT technique applied in this work for QA reinforces the power of the unsupervised representation of language learned by pretraining Transformers. Including target domain documents during pretraining consistently improves the final model performance on a domain adaptation setting. This simple, yet powerful, adaptation is capable of outperforming more complex approaches proposed on the literature, such as automatic question generation.

Future work should explore the effectiveness of long-sequence Transformers [56] [57] on the Reader such that it can perform cross attention among passages. Also, on the Retriever module, it is important to apply DAPT on DPR to study the effect of this technique on retrieval. DPR training is much more extensive than DPR-Reader, so we

suspect that DAPT will be less effective, as DPR tends to forget more of the information from pretraining than DPR-Reader. This must be investigated empirically.

REFERENCES

1 NISHIDA, K. et al. Unsupervised domain adaptation of language models for reading comprehension. In: *LREC 2020 - 12th International Conference on Language Resources and Evaluation, Conference Proceedings*. Marseille: European Language Resources Association (ELRA), 2020. p. 5392–5399. ISBN 9791095546344.

2 MANYIKA, J. et al. *Big data: The next frontier for innovation, competition, and productivity*. [S.l.]: McKinsey Global Institute, 2011.

3 ITTOO, A.; NGUYEN, L. M.; van den Bosch, A. Text analytics in industry: Challenges, desiderata and trends. *Computers in Industry*, v. 78, p. 96–107, 2016. ISSN 0166-3615.

4 TANGUY, L. et al. Natural language processing for aviation safety reports: From classification to interactive analysis. *Computers in Industry*, v. 78, p. 80–95, 2016. ISSN 0166-3615.

5 ASAKURA, T. et al. A quantitative evaluation of natural language question interpretation for question answering systems. In: *Joint International Semantic Technology Conference, Proceedings*. Awaji: Springer, 2018. p. 215–231.

6 UTOMO, F. S.; SURYANA, N.; AZMI, M. S. Question answering system: A review on question analysis, document processing, and answer extraction techniques. *Journal of Theoretical & Applied Information Technology*, v. 95, n. 14, 2017.

7 RAJPURKAR, P. et al. SQuAD: 100,000+ questions for machine comprehension of text. In: *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas: Association for Computational Linguistics, 2016. p. 2383–2392.

8 KWIATKOWSKI, T. et al. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*, 2019.

9 LEE, J. et al. BioBERT: a pre-trained biomedical language representation model for biomedical text mining. *Bioinformatics*, Oxford University Press, v. 36, n. 4, p. 1234–1240, 2020.

10 FISCH, A. et al. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In: *Proceedings of 2nd Machine Reading for Reading Comprehension (MRQA) Workshop at EMNLP*. Hong Kong: Association for Computational Linguistics, 2019.

11 KOLOMIYETS, O.; MOENS, M.-F. A survey on question answering technology from an information retrieval perspective. *Information Sciences*, Elsevier, v. 181, n. 24, p. 5412–5434, 2011.

12 ATHIRA, P.; SREEJA, M.; REGHURAJ, P. Architecture of an ontology-based domain-specific natural language question answering system. *International Journal of Web & Semantic Technology*, Academy & Industry Research Collaboration Center (AIRCC), v. 4, n. 4, p. 31, 2013.

13 DAMIANO, E. et al. Towards a framework for closed-domain question answering in italian. In: *2016 12th International Conference on Signal-Image Technology Internet-Based Systems (SITIS), Proceedings*. Naples: Institute of Electrical and Electronics Engineers Inc., 2016. p. 604–611.

14 ABDI, A.; IDRIS, N.; AHMAD, Z. QAPD: an ontology-based question answering system in the physics domain. *Soft Computing*, Springer, v. 22, n. 1, p. 213–230, 2018.

15 FADER, A.; ZETTLEMOYER, L.; ETZIONI, O. Open question answering over curated and extracted knowledge bases. In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. Barcelona: Association for Computing Machinery, 2014. p. 1156–1165.

16 BESBES, G.; BAAZAOUI-ZGHAL, H.; GHEZELA, H. B. An ontology-driven visual question-answering framework. In: *2015 19th International Conference on Information Visualisation, Proceedings*. [S.l.]: IEEE, 2015. p. 127–132.

17 GAMAL, A. et al. Improving question answering system based on a hybrid technique. *Journal of Computer Science*, v. 14, p. 1202.1212, 08 2018.

18 JURAFSKY, D.; MARTIN, J. H. *Speech and Language Processing*. 3rd ed. draft. ed. USA: [s.n.], 2019.

19 PIZZATO, L. A.; MOLLÁ, D. Indexing on semantic roles for question answering. In: *Coling 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering*. [S.l.: s.n.], 2008. p. 74–81.

20 MIKOLOV, T. et al. Distributed representations of words and phrases and their compositionality. In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. Red Hook, NY, USA: Curran Associates Inc., 2013. (NIPS'13), p. 3111–3119.

21 TAY, Y.; TUAN, L.; HUI, S. Hyperbolic representation learning for fast and efficient neural question answering. In: *WSDM 2018 - Proceedings of the 11th ACM International Conference on Web Search and Data Mining*. Marina Del Rey: Association for Computing Machinery, Inc, 2018. v. 2018-Febuary, p. 583–591.

22 QIU, X.; HUANG, X. Convolutional neural tensor network architecture for community-based question answering. In: *IJCAI International Joint Conference on Artificial Intelligence, Proceedings*. Buenos Aires: IJCAI, 2015. v. 2015-January, p. 1305–1311.

23 SOCHER, R. et al. Dynamic pooling and unfolding recursive autoencoders for paraphrase detection. In: *Advances in Neural Information Processing Systems 24: 25th Annual Conference on Neural Information Processing Systems 2011, NIPS 2011, Proceedings*. [S.l.: s.n.], 2011.

24 LUKOVNIKOV, D. et al. Neural network-based question answering over knowledge graphs on word and character level. In: *26th International World Wide Web Conference, WWW 2017, Proceedings*. Perth: International World Wide Web Conferences Steering Committee, 2017. p. 1211–1220.

25 TAY, Y. et al. Learning to rank question answer pairs with holographic dual LSTM architecture. In: *SIGIR 2017 - Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*. Tokyo, Shinjuku: Association for Computing Machinery, Inc, 2017. p. 695–704.

26 TRAN, N.; NIEDEREÉE, C. Multihop attention networks for question answer matching. In: *41st International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2018, Proceedings*. Ann Arbor: Association for Computing Machinery, Inc, 2018. p. 325–334.

27 VASWANI, A. et al. Attention is all you need. In: *Advances in Neural Information Processing Systems*. Long Beach: Neural information processing systems foundation, 2017. v. 2017-December, p. 5999–6009.

28 DEVLIN, J. et al. BERT: Pre-training of deep bidirectional transformers for language understanding. In: *NAACL HLT 2019 - 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies - Proceedings of the Conference*. Minneapolis: Association for Computational Linguistics, 2019. v. 1, p. 4171–4186.

29 RAJPURKAR, P.; JIA, R.; LIANG, P. Know what you don't know: Unanswerable questions for SQuAD. In: *ACL 2018 - 56th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference (Long Papers)*. Melbourne: Association for Computational Linguistics, 2018. v. 2, p. 784–789.

30 LIU, Y. et al. Roberta: A robustly optimized BERT pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

31 RADFORD, A. et al. Language models are unsupervised multitask learners. *OpenAI blog*, v. 1, n. 8, p. 9, 2019.

32 KARPUKHIN, V. et al. Dense passage retrieval for open-domain question answering. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 2020. p. 6769–6781.

33 LEE, K.; CHANG, M.-W.; TOUTANOVA, K. Latent retrieval for weakly supervised open domain question answering. In: *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*. Florence: Association for Computational Linguistics, 2020. p. 6086–6096.

34 JOHNSON, J.; DOUZE, M.; JÉGOU, H. Billion-scale similarity search with GPUs. *IEEE Transactions on Big Data*, p. 1–1, 2019.

35 GUU, K. et al. REALM: Retrieval-Augmented Language Model Pre-Training. *arXiv:2002.08909 [cs]*, fev. 2020. ArXiv: 2002.08909. Disponível em: ⟨http://arxiv.org/abs/2002.08909⟩.

36 YANG, W. et al. Critically examining the neural hype: weak baselines and the additivity of effectiveness gains from neural ranking models. In: *Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval*. Paris: Association for Computing Machinery, Inc, 2019. p. 1129–1132.

37 ROBERTSON, S.; ZARAGOZA, H. *The probabilistic relevance framework: BM25 and beyond*. [S.l.]: Now Publishers Inc, 2009.

38 SAKAGUCHI, K. et al. WINOGRANDE: An adversarial winograd schema challenge at scale. In: *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*. New York: AAAI press, 2020. p. 8732–8734.

39 KLEIN, T.; NABI, M. Learning to answer by learning to ask: Getting the best of GPT-2 and BERT worlds. *arXiv preprint arXiv:1911.02365*, 2019.

40 SUBRAMANIAN, S. et al. Neural models for key phrase extraction and question generation. In: *Proceedings of the Workshop on Machine Reading for Question Answering*. Melbourne, Australia: Association for Computational Linguistics, 2018. p. 78–88.

41 ALBERTI, C. et al. Synthetic qa corpora generation with roundtrip consistency. In: *ACL 2019 - 57th Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference*. Florence: Association for Computational Linguistics, 2020. p. 6168–6173. Cited By :45.

42 NEMA, P. et al. Let's ask again: Refine network for automatic question generation. In: *EMNLP-IJCNLP 2019 - 2019 Conference on Empirical Methods in Natural Language Processing and 9th International Joint Conference on Natural Language Processing, Proceedings of the Conference*. Hong Kong: Association for Computational Linguistics, 2020. p. 3314–3323.

43 LI, H. et al. D-NET: A pre-training and fine-tuning framework for improving the generalization of machine reading comprehension. In: *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*. Hong Kong, China: Association for Computational Linguistics, 2019. p. 212–219.

44 YANG, Z. et al. XLNet: Generalized autoregressive pretraining for language understanding. In: WALLACH, H. et al. (Ed.). *Advances in Neural Information Processing Systems*. [S.l.]: Curran Associates, Inc., 2019. v. 32.

45 SUN, Y. et al. ERNIE 2.0: A continual pre-training framework for language understanding. In: *AAAI 2020 - 34th AAAI Conference on Artificial Intelligence*. New York: AAAI press, 2020. p. 8968–8975.

46 TALMOR, A.; BERANT, J. MultiQA: An empirical investigation of generalization and transfer in reading comprehension. In: *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*. Florence, Italy: Association for Computational Linguistics, 2019. p. 4911–4921.

47 SHAKERI, S. et al. End-to-end synthetic data generation for domain adaptation of question answering systems. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. Online: Association for Computational Linguistics, 2020. p. 5445–5460.

48 GURURANGAN, S. et al. Don't stop pretraining: Adapt language models to domains and tasks. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. Online: Association for Computational Linguistics, 2020. p. 8342–8360. Disponível em: ⟨https://aclanthology.org/2020.acl-main.740⟩.

49 TSATSARONIS, G. et al. An overview of the BIOASQ large-scale biomedical semantic indexing and question answering competition. *BMC bioinformatics*, BioMed Central, v. 16, n. 1, p. 1–28, 2015.

50 TRISCHLER, A. et al. NewsQA: A machine comprehension dataset. In: *Proceedings of the 2nd Workshop on Representation Learning for NLP*. Vancouver, Canada: Association for Computational Linguistics, 2017. p. 191–200.

51 SAHA, A. et al. DuoRC: Towards Complex Language Understanding with Paraphrased Reading Comprehension. In: *Meeting of the Association for Computational Linguistics (ACL)*. Melbourne: Association for Computational Linguistics, 2018.

52 YANG, Y.; YIH, W.-T.; MEEK, C. WIKIQA: A challenge dataset for open-domain question answering. In: *Conference Proceedings - EMNLP 2015: Conference on Empirical Methods in Natural Language Processing, Proceedings*. Lisbon: Association for Computational Linguistics, 2015. p. 2013–2018.

53 LOSHCHILOV, I.; HUTTER, F. Decoupled weight decay regularization. *arXiv preprint arXiv:1711.05101*, 2017.

54 GOMI, E. S. et al. *Sistemas de Machine Learning para Gêmeos Digitais de Unidades Estacionárias de Produção de Petróleo e Gás - Relatório Técnico Parcial 2 - Período 14/11/2019 a 20/10/2020*. São Paulo, 2020.

55 WOLF, T. et al. Transformers: State-of-the-art natural language processing. In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*. Online: Association for Computational Linguistics, 2020. p. 38–45.

56 ZAHEER, M. et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, v. 33, 2020.

57 BELTAGY, I.; PETERS, M. E.; COHAN, A. Longformer: The long-document transformer. *arXiv:2004.05150*, 2020.