

**Cross-Lingual Transfer Learning with Adapters for Multilingual Question
Generation**

Subtitle

Silviya Silwal

MSc. Dissertation



Department of Artificial Intelligence
Institute of Linguistics and Language Technology
Faculty of Information and Communication Technology

University of Malta

September, 2023

Supervisor(s):

Dr. Marc Tanti, Institute of Linguistics and Language Technology, University of Malta

Submitted in partial fulfilment of the requirements for the degree of
Master of Science in Human Language Science and Technology (HLST)



L-Universit`
ta' Malta

University of Malta Library – Electronic Thesis & Dissertations (ETD) Repository

The copyright of this thesis/dissertation belongs to the author. The author's rights in respect of this work are as defined by the Copyright Act (Chapter 415) of the Laws of Malta or as modified by any successive legislation.

Users may access this full-text thesis/dissertation and can make use of the information contained in accordance with the Copyright Act provided that the author must be properly acknowledged. Further distribution or reproduction in any format is prohibited without the prior permission of the copyright holder.

Supervisor(s)

Dr. Marc Tanti

Institute of Linguistics and Language Technology

University of Malta

Acknowledgements

I wish to express my heartfelt gratitude to those who have supported me throughout my master's journey, especially during the development of this thesis. This accomplishment is not a just my own but a reflection of collective efforts.

I am profoundly thankful to Dr. Lucile Callebert at Synapse, who not only provided me with the opportunity to be a part of this project but also believed in my ideas from the beginning. It has been a treat to work under her guidance, and I consider her the finest mentor I've had the pleasure to work with.

Dr. Marc Tanti deserves special mention for his unwavering dedication to his students. It is rare to find professors who display such a genuine interest in their students' perspectives. I deeply appreciate the invaluable feedback and guidance he provided me throughout this journey.

To my friends who have been my pillars of strength over the past two years: I am grateful for the laughter and fun we shared. Special thanks to my LCT classmates – Andy, Averie, and Hossain – for sharing this amazing year in France with me. Together, we faced challenges, found encouragement, and created memorable travel moments.

I extend my appreciation to the QUANTUM team at Synapse, IRIT and LORIA for their feedback and support that significantly contributed to the successful completion of this thesis.

Finally, my journey would have been incomplete without the continuous support of my family and friends. Your daily encouragement, faith in my capabilities, and even the thoughtful gifts provided the motivation I needed. I look forward to reuniting with you all soon.

Abstract

In recent times, chatbots have become an integral component of online services, offering users a quick and efficient means to access information and accomplish tasks. However, creating a chatbot that can accurately comprehend and address user queries is a challenging task. To enhance chatbot performance, a promising technique is the use of question generation, which helps to align fresh user queries with existing solutions. This thesis project aims to develop an automatic question generation system that can produce relevant questions from knowledge bases, while also exploring the effectiveness of this approach in improving chatbot performance. In our research, we experiment with two distinct methods: one based on adapter training and the other leveraging the GPT-3.5 turbo model. Through various evaluations of the results generated by these systems, we find that training both language and task adapters is an effective approach for cross-lingual transfer learning, outperforming our baseline metrics. Taking into account computational resources and integration ease, we selected the GPT model for incorporation into the chatbot system, thereby enhancing its capabilities via question generation.

Contents

1	Introduction	1
1.1	Aims and Objectives	3
1.2	Thesis Outline	5
2	Company and Context of Project	6
2.1	Impact of Project	7
3	Background	9
3.1	Natural Language Processing (NLP)	9
3.2	Models	10
3.3	Natural Language Generation (NLG)	13
3.4	Pre-trained Language Models	16
3.5	Adapters and Adapter-based tuning	17
4	Related Works	20
4.1	What is a Question?	20
4.2	Traditional Approaches	21
4.3	End-to-End Seq2Seq Models	25
5	Data	28
5.1	Language Selection	28
5.2	Pre-training Data for Language Adapters	29
5.3	Fine-tuning Data for Task Adapter	31
5.3.1	Data Transformation	35
6	Methodology	37
6.1	Adapter-Based System	37
6.1.1	Overview	38
6.1.2	Adding adapters to mT5	39
6.1.3	Language Adapters	41
6.1.4	Task Adapter	43

6.1.5	Final Pipeline	43
6.1.6	Hardware and Hyperparameters	46
6.2	LLM Langchain System	47
6.2.1	Model Selection	47
6.2.2	System Architecture	47
7	Evaluation and Results	50
7.1	Baseline Models	50
7.2	Generated Question evaluation	50
7.2.1	Data	51
7.2.2	Results	51
7.3	Evaluation through chatbot performance	53
7.3.1	Data	54
7.3.2	Results	55
8	Discussion	58
9	Conclusion	63
10	Bibliography	64
A	Prompts	80
A.1	Prompt + Context	80
A.2	Prompt + Title	80
B	Examples of Generated Questions	81
B.1	Adapter-Based System	81
B.1.1	English Example	81
B.1.2	French Example	81
B.2	LLM Langchain System	82
B.2.1	Prompt + Context (English)	82
B.2.2	Prompt + Context (French)	82
B.2.3	Prompt + Title (English)	83

B.2.4 Prompt + Title (French)	83
---	----

List of Tables

1	Example from Eli5 dataset	4
2	QG dataset	35
3	Span masked language modeling	42
4	Example from SQuAD 2.0 dataset	44
5	Evaluation Context Dataset	51
6	English Evaluation Result	52
7	French Evaluation Results	53
8	English Testbook Result	55
9	French Testbook Result	56
10	English Context-Title	58
11	Examples of English generated questions	59
12	French Context-Title	60
13	Examples of French generated questions	61

List of Figures

1	RNN structure	11
2	Scaled Dot-Product Attention	12
3	Multi-Head Attention	14
4	Sequence-to-Sequence Model	14
5	Global attention Model	15
6	Sentence length	31
7	English corpora Frequency	32
8	French corpora Frequency	33
9	Unique words	34
10	QG dataset distribution	36
11	MAD-X architecture	39
12	Language Adapter	42
13	QG Task Adapter	45
14	AQG Final Pipeline	46
15	LLM Langchain Pipeline	49
16	Testbook evaluation	55

1 Introduction

Questions are an important part of our daily interactions. They do not just serve as information-gathering tools, but also express the speaker’s intentions, account for language variations, and adapt to the context [32]. As we shift from human interaction to digital communication, the ability to formulate questions becomes important for developing intelligent machines. We call this Automatic Question Generation (AQG) - the ability for a machine to autonomously generate questions from a given text or images [64].

Historically, question generation has been an important part of conversational AI systems, such as ELIZA [93], setting the stage for modern, more complex and intelligent systems. As our world becomes increasingly data-driven, there is a growing need for smart tools that can help parse, understand, and engage with the large amount of information available. AQG plays a crucial role here, transforming passive text consumption into an interactive, engaging experience. Its applications span from education, aiding in creating study materials and assessment tests [30], to customer service, assisting in the creation of dynamic and intuitive chatbots [66].

Likewise, as our world becomes more and more interconnected, the importance of multilinguality in AQG grows significantly. This widens the use of machines by enabling them to generate questions in various languages and also promotes more inclusive digital spaces [84]. However, this also introduces an extra layer of complexity due to linguistic diversity, syntactic variations, and semantic intricacies among different languages. The development of multilingual AQG models requires innovative solutions, including cross-lingual transfer learning and language-adaptive techniques, to effectively capture and reproduce the distinctive aspects of various languages [50].

AQG systems are generally put into two categories: closed-domain and open-domain. Open-domain question generation is not limited to a specific subject and formulates its questions from a wide range of topics, using universal ontologies. On the other hand, closed-domain question generation is restricted to certain domains such as education or law and relies on domain-specific knowledge to formulate its questions [64].

Over the years, considerable advancements have been made in both open and closed

domain question generation. For example in the open-domain front we have seen the use of Recurrent Neural Networks (RNNs) to generate factoid questions, and identifying important points for reading comprehension question generation. Similarly, in the close-domain front progress has been made in areas such as constituent and dependency parsing [89], representations via lexical-functional grammar [41], and rule-based setups using semantic role labeling [27].

Along with the advancements, it is also important to highlight the methodological shifts that have shaped the field. Initially, the emphasis was on designing systems using rule-based and statistical strategies, which would transform input text into questions, as indicated by the works of Heilman and Smith [38] and Rus et al. [80]. However, the landscape shifted with the introduction of neural methodologies. This shift transformed question generation as a two-step process - encoding and decoding. The encoder's role involves translating the input text into a continuous representation. Following this, the decoder generates a question from this representation, as presented by Duan et al. [25] and Du and Cardie [23].

AQG falls under the realm of natural language generation (NLG). The task of NLG has always been challenging since we need make sure that the sentence is syntactically correct while maintaining the correct intention. The evolution of NLG models has been significant, transitioning from basic rule-based systems to more sophisticated RNNs, which have demonstrated decent performance in tasks such as Neural Machine Translation (NMT). More recently, the introduction of transformer language models, including BERT [21] and GPT-3 [13] has further elevated the proficiency in text generation tasks. These developments provide us with an increasingly diverse toolkit for NLG.

Yet, despite this progress, there are notable challenges that need to be addressed. For instance, the dominance of English-language training datasets poses challenge to the training of multilingual AQG systems. Furthermore, the rise of neural methodologies and transformer models has led to a large number of new language models, but using these for AQG is tricky due to constraints like limited computing resources and training dataset availability.

1.1 Aims and Objectives

This master thesis addresses the task of creating a multilingual Automatic Question Generation (AQG) system. Our work acknowledges the constraints presented by limited multilingual datasets and the significant computational resources required for training large language models (LLMs). To navigate these constraints we propose experiment with two methods.

The first is our main proposed pipeline which includes the use of adapter modules to establish a modular framework system, allowing the development of a multilingual AQG system. The second method includes developing a system that makes use of Large Language Models (LLMs), in our case the recently released OpenAI GPT-3.5 turbo model¹, to generate question through prompting.

Our objectives is thus formally defined as follows: Given a text segment from a documentation corpus, our goal is to develop an AQG system capable of generating relevant and meaningful questions from the text excerpt. These generated questions should be answerable using the information contained within the said text.

We believe that the capacity to generate questions from a documentation corpus can drastically enhance the knowledge base and overall performance of a chatbot. By generating a large number of known questions, the chatbot can become more adept at managing new user queries, particularly those that have resemblance to existing questions. The ultimate outcome is the creation of a more efficient and effective chatbot system. More information about the chatbot is provided in Section 2.

Table 1 shows an example from the Eli5 dataset [26], which is widely recognized for its role in English QA tasks. Within this task, the aim is to develop algorithms capable of generating the correct response to a given query, such as “How do Jellyfish function without brains or nervous systems?”, taking into account the context provided in Table 1. In contrast, the AQG task reverses the idea utilizing the same context from Table 1 to guide the system in formulating a question similar to the original one. Hence, the new question could be something like, “How do Jellyfish function without brains or nervous systems?”. But instead of requiring an exact match the generated question could

¹<https://platform.openai.com/docs/models/gpt-3-5>

Context	The box jellyfish nervous system is divided into three functional parts namely; rhopalia, conducting nerve ring, and motor nerve net. [...] Jellyfish do not have brains, and most barely have nervous systems. They have primitive nerve cells that help them orient themselves in the water and sense light and touch. [...] While they don't possess brains, the animals still have neurons that send all sorts of signals throughout their body. "It is not true that jellyfish have no central nervous systems. They have an unusual nervous system," [...]
Question	How do Jellyfish function without brains or nervous systems?
Answer	Jellyfish may not have a brain, but they have a rough nervous system and innate behaviours. However, they are very simple creatures. They're invertebrate: creatures without a backbone. Most jellyfish have really short life spans. [...] As their name implies, they are largely composed of basically jelly inside a thin membrane. They're over 95% water.

Table 1: Example from Eli5 dataset [26]

be, "How can jellyfish survive without having a brain or a nervous system?" or "Can you explain how jellyfish manage their biological functions without a brain or nervous

system?”. The success criteria for this generation is twofold; the newly generated question must be understandable to humans, and its answer should be identifiable from the given context.

In relation to this thesis project specifically, our interest lies in training an answer-agnostic AQG system. This contrasts with standard AQG tasks in that the system will not receive any given answer as input during training to provide clues about what the answer should be within the context. Typically, answer-aware systems are trained by feeding the context and answer or answer span, instructing the model to generate the question based on these inputs. For this project, however, the input will simply be the context. This is because the ultimate goal is to create a system capable of generating multiple questions for each context, with each question exploring different aspects of the context rather than focusing on a specific answer span within it.

1.2 Thesis Outline

The rest of this thesis will be structured as follows:

Chapter 2 provides information about the company and the impact of the project within the company. Chapter 3 introduces some background information about the basic concepts of NLG and AQG. Chapter 4 contains literature review on various AQG systems. Chapter 5 provides information about the datasets used in this project. Chapter 6 is the main section where we present out overall methodology and system pipelines. Chapter 7 presents the evaluation methodology and results. In Chapter 8 we discuss the results and talk about the final pipeline that was integrated into the company’s system. Finally, in Chapter 9 we conclude this thesis with potential future works and an overall epilogue.

2 Company and Context of Project

This research project was developed at Synapse Développement in Toulouse, France which is a company that specializes in Natural Language Processing (NLP) and Artificial Intelligence (AI) applied to texts. It is a small to medium-sized enterprise with about twenty experts, each assigned to Technical, Communication, and Sales departments, as well as a Customer Relations and Project Management department.

Since its establishment in 1994, Synapse Développement gained recognition through its first product: Cordial, a spelling and grammar checker. It became particularly well-known because Synapse was the supplier of the French language spelling checker for Microsoft Office.

Following the success of Cordial, the company expanded its offerings and now provides three products in total:

Cordial: A spelling and grammar checker in French.

DearBot: A unique solution for conversational agents.

Semantics: A semantic analysis tool for organizations.

Synapse Développement is an ambitious company that values teamwork and innovation, which has earned it numerous awards and recognition. For instance, it received the "Entreprise de l'année" (Company of the Year) award from Occitanie Innov in 2019 and won the I-Nov competition in the Digital DeepTech category organized by BPI in 2022.

The research presented in this thesis is associated with the QUANTUM² project, which is funded by the Agence National de Recherche (ANR). This initiative is a collaborative effort involving two academic entities, LORIA from Nancy and IRIT from Toulouse, as well as an industrial partner, Synapse Développement of Toulouse. The main objective is to explore the concept of passage-level abstractive question formation. To this end, the methodology and experiments presented in this paper is related to the ones developed and conducted at Synapse Développement.

²<https://www.irit.fr/quantum/>

2.1 Impact of Project

Synapse’s DearBot chatbot solution currently operates on a hybrid system that combines rule-based methodologies with simple machine learning techniques, such as topic detection and modeling. As advances in language generation and chatbot technologies continue to unfold, Synapse is strategically positioning DearBot for the future. The roadmap for its development involves transitioning from traditional methods to leveraging Large Language Models (LLMs), expanding its operational capabilities beyond the French-speaking market to support multiple languages, and enhancing the knowledge base system integral to the DearBot framework.

The project on enhancing the knowledge base is led by Dr. Lucile Callebert. In the current pipeline of the DearBot solution, client data undergoes a preprocessing phase. This phase involves the extraction of information from client documents, spanning formats like HTML and PDF. It identifies contexts and titles within these documents. Once this pre-processing is done the QG system uses the contexts or titles to formulate relevant questions. Once the questions are formulated, the contextual data, paired with its associated questions, is stored in the knowledge base.

The main role of the knowledge base is to link user queries with the right questions. This ensures users get accurate answers quickly. The quality of questions derived from client data is critical for DearBot’s efficiency. The QUANTUM research project is focused on finding new ways to generate questions automatically after the pre-processing step. The goal is to improve the knowledge base, leading to better performance by DearBot.

The entirety of the project involved gathering and prepping training datasets, investigating different techniques, designing specific pipelines, evaluating their results, and ultimately choosing the best one for integration within the DearBot system. These steps were undertaken with the challenges of limited training data (as discussed in the previous chapter), constraints on resources, and time restrictions in mind. We go into detail about the training dataset in Chapter 5 and information about the hardware resources in Chapter 6. The main software and tools used at the company during the development of this project includes:

- Libraries, Framework and APIs
 - PyTorch
 - Huggingface API
 - Adapter transformer
 - OpenAI API
 - Langchain
- Tools, Integration, CI/CD and IDEs
 - Git + GitLab
 - PyCharm, Visual Studio and Visual Studio Code
 - MySQL Workbench
 - Postman
- Project Management
 - Jira, SCRUM

3 Background

In this chapter, we dive into a detailed analysis of question generation, covering both theoretical and applied aspects. The content is divided into three key sections. Firstly, we will look through the foundations of Natural Language Processing (NLP). Secondly, we familiarize ourselves with Natural Language Generation (NLG) and highlight a few models currently employed. Then, we examine the use of adapters in training Large Language Models (LLMs) and their significance in our task. Finally, we talk about Langchain and its capabilities with LLMs.

3.1 Natural Language Processing (NLP)

Natural Language Processing (NLP) is a subfield of artificial intelligence focused on interacting with human language data. The primary goal is to enable machines to interpret textual content and bridge the gap between the raw text and the anticipated outcome.

In an NLP model, the transformation of the input text to the desired outcomes requires turning the meaning of the text into a format that is compatible with mathematical computations. In NLP tasks, the properties used serve as the fundamental evidence linking aspects of the text to the target. There are numerous methods to create these properties. For example, tagging based on parts-of-speech [11] or recognizing specific named-entities in the text [72] can be used to examine each text section and create rules that select the combinations most connected to the target. These handcrafted features can be extremely useful for certain tasks [7]. However, creating these features requires the use of qualified individuals and can be a time-consuming process.

The challenge of automatically generating features for language processing requires that words — the fundamental units of meaning in language — be represented numerically. While simply assigning integer values to words could be one approach, the natural order of integers could unintentionally introduce bias. Thus, the one-hot encoding method was proposed, providing a vector representation for each word, where only a single element holds the value 1 and everything else has the value 0. Despite being effective in a variety of tasks, the method has its limitations. It considers each word as a separate category,

leading to long vectors equivalent to the size of our vocabulary library. A big disadvantage of this method is its size which ultimately affects memory and speed. Moreover, this method does not facilitate the comparison of similarity between individual words.

The TF-IDF matrix was introduced to mitigate this limitation. By merging term frequency [42] with inverse document frequency [77], it allocates scores to words based on their frequency in one document and across multiple documents. Hence, if there is a small cosine distance between their TF-IDF vectors, it suggests that the two words could be semantically related. Implementing methods like SVD [43] for reducing dimensions helps in preserving similarity with smaller word vectors.

Yet, the TF-IDF matrix encounters challenges similar to standard embeddings. Alterations in the corpus, for instance, adding or deleting a document, mandate a matrix overhaul. To counter this, deep learning emerged as a solution. Mikolov et al. [59] suggested an approach where the neural network, upon receiving a word vector, predicts its neighboring words. This method allows for the representation of all vocabulary words with extensive corpus training.

Advancing this idea, the GLoVE word model [67] is notable. It fuses the TF-IDF matrix's principles with prediction-centric methods, crafting a matrix that tallies the occurrences of two words co-existing within a set window. This technique notably boosts the efficacy of NLP tools.

Still, it is important to note that while these word representations, often pre-trained on large corpora, are fixed, they do not capture context-specific nuances. Peters et al. [68] suggested adding contextual knowledge to word vectors using a deep learning model, the bi-directional LSTM. Devlin et al. [21] team extended this idea to develop the BERT model, which uses Transformer model [90] components for contextual embedding. Presently, a host of pre-trained language models (PLMs) like BERT [21], mT5 [95], and GPT-3 [13] offer powerful tools for word representation across various NLP studies.

3.2 Models

Models serve as the link between text input and the desired outcome. Traditional NLP models typically rely on feature-based methods. These custom features reflect the expert's

understanding of the task at hand. Models like the Maximum Entropy Model [9] receive feature values as inputs and yield predictions as outputs. Internal model parameters can be learned through the utilization of machine learning algorithms.

However, with the rise of deep learning and neural networks, there has been a shift towards deep learning models for solving NLP tasks. Popular models like the Recurrent Neural Network (RNN) [59], and Long short-term memory (LSTM) [29], have emerged as key players. These architectures integrate the word embeddings of a sentence sequentially, facilitating the transfer of contextual data. The RNN-centric model encodes and outputs the input text at its final unit, as depicted in Figure 1.

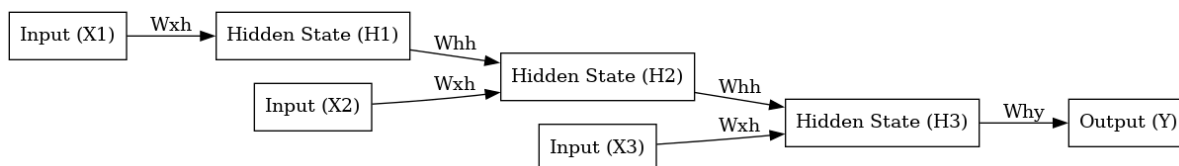


Figure 1: RNN structure

Models built on Recurrent Neural Networks (RNNs) have seen significant success in processing sequential data, but suffer from an inherent flaw - the inability to maintain long-term dependencies. The other models were introduced as possible solutions to this issue, however, they only partially address it. The Transformer architecture, proposed by Vaswani et al. [90], currently stands out as the most viable answer to this dilemma.

Within the Transformer framework, the recurrent process is replaced by a self-attention technique deployed during both the encoding and decoding stages. This attention method functions by giving weights to different features, allowing those with higher weights to play a more decisive role in the final result.

Each element of the input (every matrix column vector) is run through three feed-forward layers, leading to the creation of three distinct outputs termed as query, key, and value. This design promotes simultaneous computation via matrix operations, producing trio matrices for query (Q), key (K), and value (V).

In the realm of self-attention, each vector's fresh depiction in the query matrix undergoes a dot-product operation with each segment of the key matrix. The output value is then channeled through a softmax layer, which designates the weight (attention) of

the complete sequence for that specific vector. The weighted aggregate value depictions is then used as a refreshed representation for the original vector. This methodology is executed for all initial vectors, culminating in a definitive output matrix post one self-attention cycle. The vertical and horizontal dimensions of this output matrix align with the sequence duration of the input and the hidden size of the sequential layer respectively.

A simplified computational flow can be visualized in Figure 2.

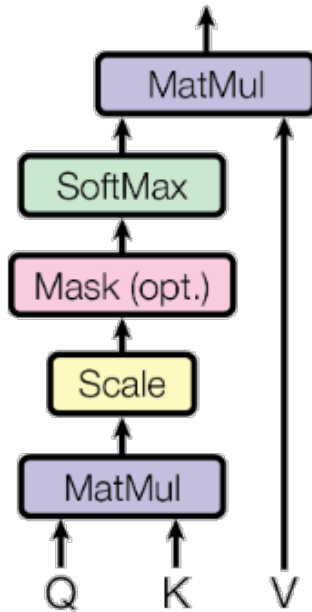


Figure 2: Scaled Dot-Product Attention [90]

All these computations can be concurrently performed through matrix manipulation, formulated as,

$$Attention(Q, K, V) = softmax \left(\frac{QK^T}{\sqrt{d_k}} V \right), \quad (1)$$

Here, Q, K, V denote the matrices for the query, key, and value respectively. The modifier d_k represents the size of each query, intended to prevent the attention outputs from becoming overwhelmingly high.

Following the creation of this new representation matrix, the final depiction of a single self-attention block is derived after an 'add and norm' layer. The addition procedure resonates with the principles of the widely acknowledged residual block in computer vi-

sion [34]. This method combines the original sequence representations with the fresh self-attention outputs, countering potential data loss in deeper layers. The term 'norm' signifies the layer-normalization method, which streamlines inputs across features [6]. Within the Transformer's context, this procedure is standardized across each sequence's tokens.

Furthermore, the Transformer architecture introduces a multi-head attention system to embrace varied dimensions of the sequence [90]. Each head comprises distinct sets of (query, key, value). The illustration in Figure 3, combined with the add and norm layer, constitutes the Transformer's foundational blocks. The overall model is constructed by layering these blocks. In contrast to RNNs, which use different gates to conserve context, the Transformer's self-attention method lets every term to receive information from any part of the sequence, crafting a new representation vector for said term. Additionally, by sidelining the traditional RNN architecture, operations become good for parallelism.

Yet, a notable drawback is that matrix operations neglect the positional context of each input representation, so a positional encoding is needed before the first multi-headed attention block. The pioneers of [90] used sine and cosine functions to produce a rhythmic positional encoding. However, for the BERT architecture, positional encodings are adaptively learned alongside other embeddings [21].

3.3 Natural Language Generation (NLG)

Answering question merely requires a straightforward response, whereas crafting questions calls for a language generation process. This task of natural language generation (NLG) is intricate because it demands both upholding the message and formatting the sentence with correct grammar. A noteworthy application of this is the Neural Machine Translation (NMT), mainly utilizing the encoder-decoder framework, first presented by Allen [5]. This design is generally seen as the decoder formulating each word based on a linguistic template that not only considers the previous word but also integrates encoded context. The model portrayed in Figure 4 offers a basic sequence-to-sequence representation. The value of this design is its proficiency in holistic comprehension and facilitating text generation. The attention mechanism can also be integrated with the sequence-to-sequence

Multi-Head Attention

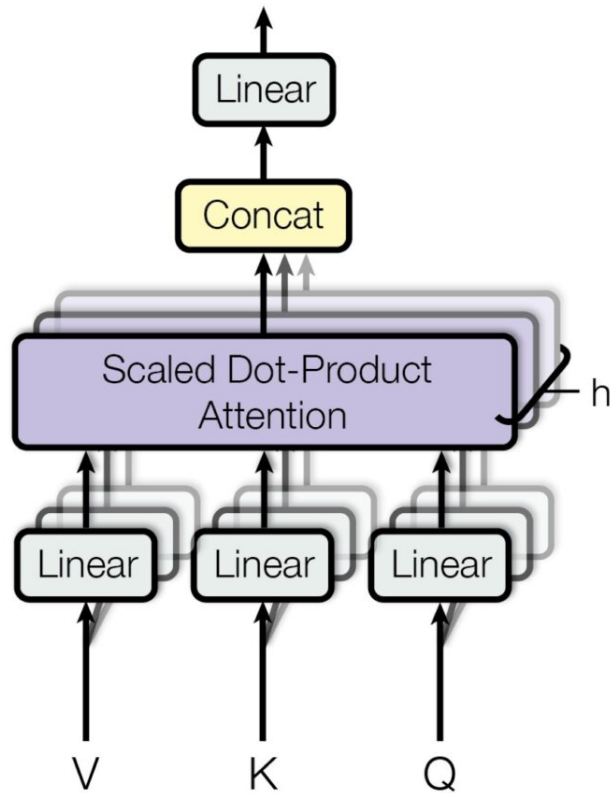


Figure 3: Multi-Head Attention [90]

model.

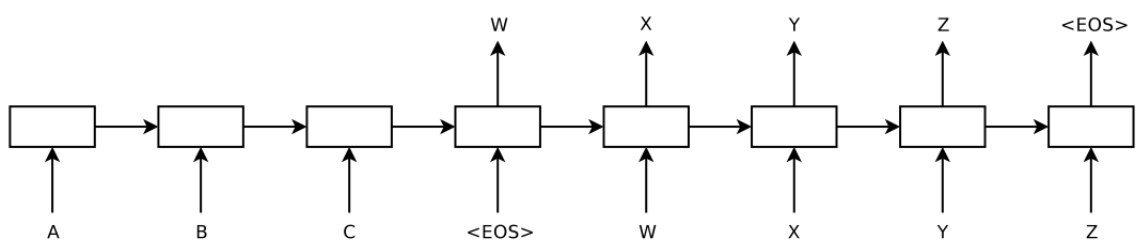


Figure 4: Sequence-to-Sequence Model [87]

Figure 5 demonstrates that Luong et al. [57] proposed a model adding supplementary data to each of the decoder's recurrent units, as opposed to solely feeding encoded details to the primary recurrent unit.

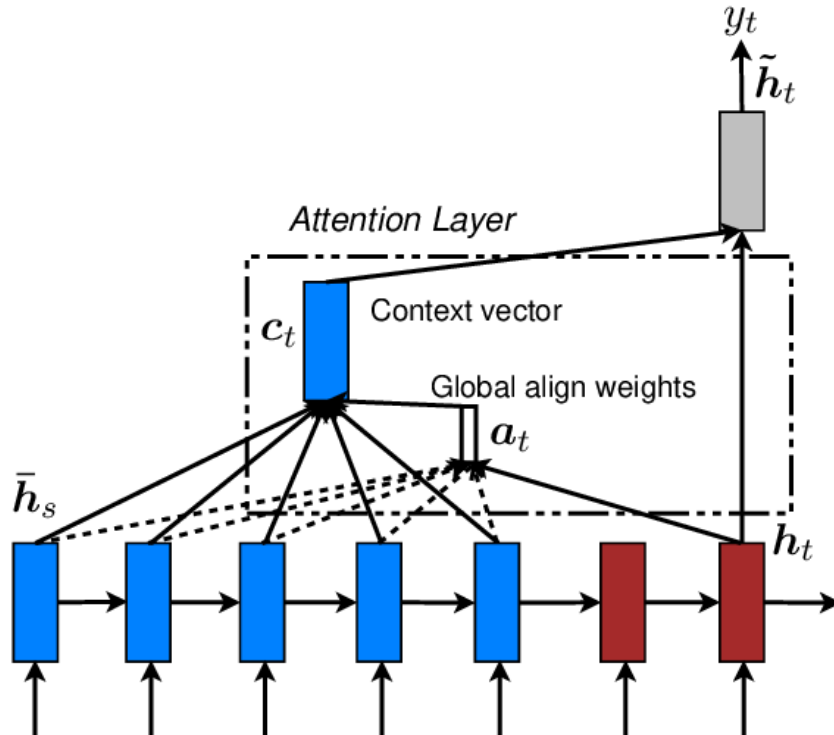


Figure 5: Global attention Model [57]

The methodology is distinct during the training period, where the teacher-forcing strategy can be applied. When generating each word step-by-step during the inference phase, the most direct application is the greedy search, always choosing the word with the top likelihood of appearing next. Yet, since the model's efficiency is judged by the sequence quality, the aim should be to choose the sequence with the most promising likelihood. Given that the complete sequence is not apparent during generation, the word with the peak probability, based on previously generated words, is not always part of the most probable sequence. Thus, the beam search technique has been introduced. In the beam search, the top-k words with highest probabilities are tracked, and the resulting sequences are re-evaluated based on their probability, derived from the multiplied probabilities of each word in the sequences. While the beam search demands more computational resources, it outperforms the greedy search approach.

3.4 Pre-trained Language Models

Recent advancements in pre-trained language models, including BERT [21], mT5 [95], and GPT-3 [13], have significantly improved various NLP tasks. BERT, or Bidirectional Encoder Representations from Transformers [21], uses the Transformer model’s self-attention mechanism, as described in section 3.2. It stands out because of its many layers of self-attention and a larger hidden unit size. Its strong performance is due to training on a large amount of text.

BERT is trained in two main steps. Initially, select words are masked, prompting the model to predict these words based on surrounding context. Simultaneously, it learns to determine if a sentence logically succeeds another using Next Sentence Prediction (NSP). This dual approach outlines BERT’s semi-supervised training approach.

An interesting feature of BERT is that its final layer can be changed to suit different tasks, often achieving better results. Architectures like BERT, which channel knowledge acquired during initial training to downstream tasks, are called pre-trained language models (PLMs). This ability to use learned knowledge in various tasks means many NLP applications can be built on top of BERT, making the process simpler.

Alongside BERT, the GPT series from OpenAI has become very popular. Starting with the original GPT, the series has expanded with GPT-2 [73] and GPT-3 [13]. Unlike BERT, which is a discriminative model trained to predict masked words, GPT models are generative in nature, primarily focusing on predicting the next word in a sequence. They use a unidirectional, left-to-right Transformer architecture during training. GPT-2, for instance, was notable for its ability to generate coherent and contextually relevant long-form text. GPT-3, on the other hand, scaled up in terms of model parameters and demonstrated remarkable capabilities not only in text generation but also in zero-shot and few-shot learning. An interesting development of the GPT series is ChatGPT, made to simulate human-like conversations.

A key feature of GPT models is their use of “prompts” to guide their answers. While the GPT models emphasized the use of “prompts” to guide their outputs, it is worth noting that the notion of using prompts or seed inputs in computational modeling is not new. However, because GPT models can generate such detailed responses, prompts have

become key in directing their outputs, making the models stand out in many NLP tasks.

In simple terms, a “prompt” is a starting clue or question given to a model, similar to how we might start a conversation. For models like GPT, prompts are vital. For instance, if we ask the model, “Tell me about france,” it will provide information about france. What is impressive about models like GPT-3 is that even a vague prompt can lead to detailed and relevant answers. These prompts do not just start the response; they shape the whole conversation, guiding the model in the direction we want. So, when we use these language models, the way we phrase our prompt can influence the kind of information the model generates.

3.5 Adapters and Adapter-based tuning

As we have observed the growing use of pre-trained language models (PLMs) and the rise of the pre-training-fine-tuning approach, we have also seen that bigger models consistently perform better. However, as the scale of these models expands, the associated computational and maintenance costs rise significantly, often making them impractical. This issue has led to the emergence of a new line of research, centered around achieving more efficient parameter adaptations in PLMs. By fine-tuning only a subset of the model parameters, it is possible to mitigate both computational and storage challenges without compromising the power of large-scale models.

Adapters offer a solution in this context. They are small layers that are stitched into pre-trained language models which enable us to efficiently train and share new task-specific models. During training, only the parameters of the adapter layers are updated, while the parameters of the pre-trained models remain frozen.

Adapter tuning was one of the first methods for guiding pre-trained models with a restricted number of parameters. This method involves the insertion of bottleneck adapter modules between the PLM layers. During fine-tuning, only these added modules are updated. The essence of adapter-based tuning lies in its modular approach, incorporating miniature neural networks—referred to as adapters—within the transformer layer. While there exists a range of adapter structures, a basic model [40] has shown excellent results and is frequently used in research. This adapter model employs a two-fold process: a

down-projection, which refines an input feature into a reduced dimensional space, followed by an up-projection, restoring the feature to its original dimensionality. Notably, a residual connection is maintained throughout for stability.

In each segment, adapter units are introduced independently after specific network sublayers, reducing the number of tuneable parameters per layer. Usually, only about 0.5-8% of the total model [40] parameters are adjusted in this process.

While an adapter incorporates fewer adjustable parameters compared to typical fine-tuning, some methods seek to economize further by reshaping the adapter’s intrinsic layer. One such technique is by Compacter [39], which uses a mix of complex multiplication and parameter sharing. This method decreases the parameter complexity inherent in a standard adapter layer without compromising its performance. Moreover, a basic decomposition of the linear layer appears to match the performance of the original adapter layer.

Adapter-based tuning, being an addition-based method, offers the benefit of adding multiple adapters to a pre-trained model all at once. This is useful because, for instance, multi-task learning [86, 45] can take advantage of adapter-based methods. By integrating the adapters alongside the self-attention module, the pre-trained language model (PLM) can demonstrate enhanced outcomes in multi-task settings. Instead of directly applying multi-task learning on adapters, AdapterFusion [70] initially sets up task-dedicated adapters and merges the representations of these trained adapters. This process can increase the performance of transfer learning.

When it comes to computational efficiency, training adapters can be 60% quicker than regular fine-tuning, while inference is only marginally slower. Additionally, computational cost can be further reduced by removing adapters from lower transformer layers [79]. Research also suggests that adapter-based fine-tuning is more robust than traditional fine-tuning, offering better performance in few-shot and cross-lingual [35] scenarios and being more resistant to adversarial attacks [33].

Regarding computational efficiency, the training of adapters can achieve speeds up to 60% faster than conventional fine-tuning. In contrast, the inference phase is only slightly prolonged. Furthermore, computational costs can be trimmed by excluding adapters

from foundational transformer layers [79]. Studies have also indicated that adapter-based fine-tuning showcases greater robustness than its traditional counterpart (fine-tuning), delivering better results in few-shot and cross-lingual [35] contexts, and exhibiting increased resistant to malicious attacks [33].

In summary, adapters represent an efficient, task-specific extension to PLMs. While the practical integration of adapters presents challenges, platforms like AdapterHub³ provide promising avenues for their inclusion in transformer models.

³<https://adapterhub.ml>

4 Related Works

4.1 What is a Question?

Various taxonomies have been proposed to categorize questions. Lehnert’s classification [51], which classified questions based on the concept of conceptual categorization, was a significant contribution in this direction. His theory suggests that correctly understanding a question requires categorizing it into the appropriate conceptual category and failure to do so may result in incorrect reasoning. The proposed classification system includes thirteen categories, such as causal antecedent, goal orientation, enablement, causal subsequent, verification, disjunctive, instrumental, concept completion, expectational, judgmental, quantification, feature specification, and request Lehnert [51], Mulla and Gharpure [64]. Similarly, another classification [32] analyzed questions asked during tutoring sessions. Identifying the type of question a system can generate is very important as questions can be classified based on the expected cognitive level of the answer.

Mulla and Gharpure [64] provide a classification of questions based on research in the field of question generation. The categories are: factual questions, which require a single word or group of words as the answer and can be answered by selecting a sentence from a paragraph; multiple sentences spanning questions, which require information from multiple sentences; yes/no questions, which require a Boolean response and higher-level reasoning; and deep understanding questions, which require an inference mechanism and information from various parts of a text.

In their discussion of the task of AQG, Mulla and Gharpure [64] formally define the problem as finding a function $f(I, A) = Q'$ that generates a question Q' semantically equivalent to the labelled question \mathbf{Q} , given an input I represented as a vector of relevant features, either an image or text. The authors further explain that the choice of dataset, question generation strategy, and question generator model depend on the question type, and can range from text-based rule-based approaches to deep-learning-based approaches that use appropriate representations for training the model.

Broadly speaking there are four main strategies used to handle the problem of Question Generation (QG): syntax-based methods (as suggested by Heilman and Smith [36] and Rus

et al. [80]), those based on semantics (Sag and Flickinger [81]), template-based methods (Chen [17]), and models that use sequence-to-sequence learning (Du et al. [24]; Yuan et al. [98]).

The syntax-based approach seems to work well for shorter sentences, but it does not always ensure proper grammar [96]. Semantics-based methods use specific identification techniques, such as semantic role labeling [52], while the use of templates can be very useful in certain areas where humans are needed to create strong and reliable templates [96].

However, when it comes to sequence-to-sequence learning techniques for question generation, they operate more freely. These strategies are not bound by the typical human-defined rules which are designed to change statements into questions [25].

4.2 Traditional Approaches

In recent times, the field of NLP has seen an increased focus on question generation, particularly following the introduction of the question generation shared task evaluation [80]. This event was divided into two phases: the primary phase centered on formulating questions from entire paragraphs, whereas the second phase—which attracted a higher degree of attention—concentrated on formulating questions from individual sentences.

In light of this event, many experts proposed different methods to address the challenges. For the second phase, a 2010 study Ali et al. [4] proposed a method that began by converting intricate texts into clearer sentences. Later, each sentence was labeled for suitable question format by analyzing its subject, verb, object, and preposition. However, this approach did not factor in semantic elements.

Several approaches have emerged to address the challenges of question generation. A study by Mitkov and Ha [60] aimed to formulate multiple-choice questions from digital educational content while also producing semantically relevant distractors. Their methodology encompassed the use of a corpus, WordNet, a basic parser, automated term extraction, and word sense determination. They streamlined the process by forming questions from declarative sentences, applying specific transformation guidelines.

In another research effort, the challenge of assessing vocabulary was explored. Such

evaluations often entail labor-intensive processes that involve handcrafted methods and can exhibit subjectivity [12]. A system was introduced by Brown et al. [12] to automate the formulation of vocabulary assessment queries. These queries can fall under six categories and may appear in diverse formats such as word bank or multiple-choice, with a focus on ensuring question precision.

Drawing from pre-existing medical literature templates, Wang et al. [92] devised a system to autonomously construct questions, intended to measure a reader’s comprehension of the content. The generated questions were fact-based, though they might not delve deep because only one sentence served as the foundation for each query.

Building on prior research on question generation from narrative passages [62], Chen [17] further refined the tool, focusing on enhancing it for informational texts. Their intent was to bolster the comprehension of these materials for children in the first to third grades.

In a unique approach to the QG challenge, Kalady et al. [44] incorporated syntax-based methods emphasizing keyword structures. Using named entity detection, text parsing, and spotlighting essential document phrases (U-Keys), they created fact-based and definitional questions, leading to both affirmative/negative and Wh-type questions.

Liu et al. [55] crafted a system to autonomously produce prompting questions, assisting students in crafting literature reviews. Initially, they pinpoint citations and pivotal content from student compositions. Following this, via rule-oriented strategies, they classify these citations and mold questions based on specific templates.

Contributing to the CODA initiative, Piwek and Stoyanchev [71] zeroed in on transforming monologue into dialogues, seeing question generation as a vital component for enhancing comprehension of educational materials.

In a different study, using template-driven approaches, Heilman and Smith [38] introduced a method that generated an excess of questions before ranking them. They used manually crafted rules to morph statement sentences into queries and applied a logistic regression framework for their sequencing. Additionally, Heilman and Smith [37] tackled the QG challenge related to facts in complex textual content. They proposed a method to extract clear and precise factual assertions, highlighting their technique’s heightened efficiency in formulating fact-based questions compared to standard text reduction algo-

rithms.

Agarwal et al. [3] approached the QG challenge using discourse indicators. Their approach consisted of two phases: pinpointing content and framing the question. The process begins by identifying an appropriate sentence to craft a query from. They then evaluate discourse connectors to determine the suitable question format, culminating in a syntactical transformation to sculpt the final query.

On another note, Becker et al. [8] concentrated on generating quiz-type questions from digital documents, aiming to aid independent learners in navigating new topics. A pivotal challenge was selecting the right portion of a sentence to probe. Their findings indicated that all semantic roles could be potent contenders for these spaces, culminating in the successful crafting of a Cloze (fill-in-the-gap) query maker.

Simultaneously, some researchers like Agarwal and Mannem [2] and Kumar et al. [48] concentrated their efforts on the generation of cloze questions or gap-fill questions. This involves creating questions with one correct answer and three different distractors. Using syntactic and lexical features, Agarwal and Mannem [2] identified informative sentences from the content. This was followed by choosing an appropriate key to form a question about, and lastly selecting a distractor based on its similarity to the key. Mirroring this approach, Kumar et al. [48] unveiled a method of ranking sentences, drawing on topic distribution from a thematic model. They used human annotations to train a classifier for gap selection and applied word2vec language model probabilities and the Dice Coefficient to find suitable distractors.

Presenting a fresh perspective, Zheng et al. [100] launched the K2Q platform that morphs keywords into queries. The mechanism factors in both prior queries and feedback from users. It yields a cluster of potential questions along with a compilation of refining terms. Users are empowered to opt for either a pertinent question or a refining term, post which the platform rolls out another set of queries and refinement options. This cycle persists until the user zeroes in on a preferred question or opts out.

Question generation applications can be particularly effective in enhancing the level of engagement readers experience when going through a news piece. The key idea is to embed various inquiries within the article, prompting readers to contemplate and thus

pay more attention to the subject matter. Rokhlenko and Szpektor [78] devised a system for the generation of comparable questions within news pieces. This system functions in two distinct parts - an offline and an online stage. The former stage involves constructing a collection of analogous question templates, while the latter leverages these templates to craft questions in alignment with the article’s context. The algorithm is designed such that it populates the chosen template with entities from the article to ensure a logical comparison between these entities.

Taking a slightly different route, Lindberg et al. [52] utilized a method that leverages semantic information along with a template-based strategy for producing natural language questions, to facilitate online learning. In the same vein, Mazidi and Nielsen [58] harnessed a semantic approach for formulating questions that offer various levels of complexity and are appropriate for independent study or tutoring. Their method hinges on semantic role tags to devise both questions and corresponding answers derived from a text. They spotlighted three linguistic hurdles: discerning negations, untangling co-references, and orchestrating verb structures. Yao [96] also suggested an approach rooted in semantics, with a central concept based on Minimal Recursion Semantics (MRS) transfer. Their method begins with breaking down complex sentences into simpler parts using MRS, then transforming the semantic representation from declarative to interrogative sentences, and finally crafting questions using deep grammar rules.

Labutov et al. [49] developed a mechanism to formulate questions without a thorough semantic grasp of the content. Central to their approach is using a simplified ontology tailored for specific text portions. By sourcing a set of fitting question templates resonating with the document’s characterization, they ensured broad adaptability of these templates across diverse texts. The final output is then sequenced by its pertinence to the initial data presented.

By utilizing information on named entities and predicate argument structures associated with the text of interest, Chali and Hasan [16] were able to formulate questions. They employed Latent Dirichlet Allocation to gauge the significance of the generated questions. This methodology involved identifying minor themes within the texts and measuring their similarity with the questions via an Enhanced String Subsequence Ker-

nel. To determine the structural accuracy of the produced questions, they implemented syntactic tree kernels. Wrapping up their process, they sequenced the questions according to their structural integrity and overall relevance.

Taking a broader perspective, Chali and Golestanirad [15] introduced a framework that derives detailed questions from text chunks, conscious of the fact that each textual dataset corresponds to a specific theme. They crafted 265 templates coupled with 350 guidelines for the creation of questions. The grammatical accuracy of the generated questions was evaluate using tree kernel methodologies. Likewise, they assessed the weight of the created questions, arranging them by leaning on community-driven question response systems.

4.3 End-to-End Seq2Seq Models

The effectiveness of deep learning approaches in various NLP tasks have led to numerous recent studies targeted at using these strategies. These techniques have demonstrated potential in areas such as machine translation [18], text abstraction [82], response to inquiries [85], and reading comprehension [65].

The first implementation of a deep sequence-to-sequence model to the task of question generation was done by Du et al. [24]. This strategy showed superior performance compared to the then state-of-the-art rule-based systems. In the wake of this, a variety of sequence-to-sequence models have been applied to question generation task [97, 46, 91].

Research recently conducted by Serban et al. [83] tackled the task of generating questions using Neural Network principles. By transforming knowledge graph facts into questions, they were able to produce a fact-based question-answer dataset. They utilized a RNN design to show that their system surpassed the performance of the projected template-based standard.

The fusion of natural language and computer vision domains has seen significant progress in the past few years. Tasks like generating captions for images, transcribing videos, and responding to queries about an image are some of the illustrations. Mostafazadeh et al. [61] expanded this scope by introducing a task that required a system to form a question based on a provided image.

Moreover, the process of generating questions can be combined with its counterpart,

answering questions, to improve outcomes. Tang et al. [88] perceived the generation and answering of questions as closely related functions and simultaneously optimized their models. They utilized the inherent correlation between these tasks in their training approach.

Song et al. [85] introduced a model designed for both formulating and responding to questions. Using a focused attention encoder-decoder architecture, which was guided by the desired answer, they processed both the text passage and the target response as inputs. This model then delved into understanding the query to identify deeper associations between the content and the anticipated answer. They further applied a policy gradient technique to adjust biases observed during model training and integrated mechanisms for copying and maintaining consistency in their decoding process.

Recently, Blšták and Rozinajova [10] introduced a method for generating factual questions from unstructured text. Their approach combines classical linguistics strategies based on sentence patterns with various machine learning methods. They extracted lexical, syntactic, and semantic information from the input text and used a combination of supervised and reinforcement learning to generate the questions.

With the increasing popularity of Graph Neural Networks (GNN), significant progress has been made in representation learning. The application of these networks to capture the relationship between words in context has been investigated in the QG task. Liu et al. [53] applied a Graph Convolutional Network (GCN) on a dependency parsing tree to predict clue words, which were encoded with other features using an RNN-based encoder. Chai and Wan [14] used GNNs to obtain new representations through interaction with the answer and passage.

An interesting approach was proposed by Liu et al. [54], where they focused on emulating the human question-asking process. Their model incorporated answer clues into the question generation system. They utilized syntactic parsing and chunking techniques on the input text to identify the most relevant clues for a target question. In addition, they incorporated question style rules into their framework and employed a BERT-based question answering model to identify and remove irrelevant questions.

Similarly, Zhou et al. [101] applied an attention-based encoder-decoder structure that

took into account the location of the answer in the source sentence, lexical characteristics, and the entire text as encoder inputs. The lexical elements they added encompassed part-of-speech (POS) and named entity recognition (NER) markers.

A recent study on question generation was conducted by Du et al. [24]. Their primary interest revolved around reading comprehension challenges, which involved generating questions from singular sentences as well as full paragraphs. Their choice was to adopt an attention-driven sequential learning framework. Progressing further, they augmented this model by refining the input sources. Diving deeper, in their research Du and Cardie [22] put forth a strategy to identify specific sentences within a section apt for question generation, and harnessed these sentences as the groundwork for their preliminary question generation system. To realize this, they hinged on a hierarchical neural sentence categorization approach and melded both aggregation and convolution methods for encoding.

Yuan et al. [98] also contributed to the field with their work. They proposed a method for generating questions from textual documents by combining supervised learning with reinforcement learning. Notably, they applied policy gradient methodologies to refine their model and amplify the incentives that assess the quality of the generated questions. Their overarching aim in developing the question generation framework was to amplify the performance of question response systems.

5 Data

As mentioned in Section 1, the lack of non-English datasets is a significant barrier for multilingual AQG. Moreover, datasets for question generation often come directly from question answering (QA) datasets. Typically, these QA datasets already contain the question, the answer, and the context. Therefore, rather than using the question to discover an answer, we can use the context and answer to generate a new question.

Additionally, most of the multilingual datasets we have access to are simply translations of English versions. The issue is that if these are used to train models, the task can become more of a translation task.

The data sources used in this thesis are twofold: one for pre-training and the other for fine-tuning. The Wikipedia corpus is initially used to train language-specific adaptations. Second, we modify and utilize English QA datasets to fine-tune our task-specific QG adapter. After training our models, we proceed on to the evaluation phase. Here, we utilize client data provided by Synapse Development. This data allows us to evaluate a chatbot’s performance on our models, ensuring that we have a complete understanding of its real-world applicability.

It is essential to keep in mind that the manner in which a task is framed can affect the structure of the data, and that the type of data used can affect the methods selected. So, in this chapter, we are going to provide a deeper analysis of the data and datasets we are working with.

5.1 Language Selection

Our thesis work focuses on adapter training in two languages: English and French. We use language adapters, pre-trained on these two languages, and a task adapter, fine-tuned only in English.

We selected English and French for pre-training the language adapters mainly because of two reasons. First, we had time constraints. Training language adapters on a large corpus takes significant time and resources, so we had to limit our focus to just two languages. Second, the majority of the data provided by Synapse Development, for chatbot

performance evaluation is in these two languages, which further motivated our choice.

When it comes to the task adapter, we faced a different issue. We did not have enough French data for the Automatic Question Generation (AQG) task. Therefore, we had to limit the fine-tuning of our task adapter to English Question-Answering datasets. This is in line with our methodology (see Section 6) and therefore was not a major problem.

So, while both English and French are used for pre-training the language adapters, only English is used for fine-tuning the task adapter. This setup allows us to explore cross-lingual transfer learning in a realistic yet constrained setting.

5.2 Pre-training Data for Language Adapters

The pre-training Wikipedia corpus is obtained from the Leipzig Corpora Collection [31] – Woetschatz Deutsch website⁴.

The Leipzig Corpora Collection [31] is an online database that provides free access to 136 monolingual dictionaries. Each dictionary is supplemented with statistical information to increase its usefulness. The collection includes corpora from numerous languages, all of which are formatted uniformly and obtained from parallel references. Users can simply download these resources as plain text files and, if needed, import them into a MySQL database using the provided script.

These resources have been designed for a variety of applications, tailored not only to corpus linguistics researchers but also to knowledge extraction software. All corpora share a common structure and comparable attributes and content properties. Each corpus includes randomly chosen sentences in its respective language, ranging from as few as 10,000 to as many as one million sentences. The corpora are carefully curated from sources such as newspaper articles and a variety of randomly selected web texts.

The texts within each corpus are broken down into sentences, with care taken to eliminate non-sentence elements and foreign language content. Given the importance of word co-occurrence in many applications, this data too has been preprocessed and incorporated. This means that for each word, the most relevant neighboring words are included, whether they are immediately adjacent or within the same sentence. The language statis-

⁴https://corpora.uni-leipzig.de/en?corpusId=deu_news_2022

tics website⁵ for the Leipzig Corpus Collection provides additional information about the file format and content.

For the purpose of this thesis, we focused on the English and French Wikipedia corpora, each comprising one million sentences. The English corpus⁶, sourced from the 2018 version, and the French corpus⁷, sourced from the 2021 version, were the most recent ones available on the website at the time of our research. Upon downloading the corpus, we obtained a zip folder containing various files - from metadata information and sources to SQL scripts, and more. Despite the comprehensive nature of the contents, our primary interest lay in the text file featuring the million sentences.

Analyzing sentence length distribution in our corpora is key to data preparation for model training. It affects span masking, training speed, and model performance. As seen in Figure 6, in our English corpus most sentences are around 13 words long, and approximately 50,000 sentences (5% of the corpus) fit this profile. In the French corpus, the average sentence length is 18 words, with around 42,000 sentences (4.2% of the corpus) of this length. This suggests our models will mainly train on moderate-length sentences, balancing learning context and computational efficiency.

There are also shorter (4-5 word) and longer (45-49 word) sentences in the corpora, but these instances are extremely rare. Short sentences might make learning harder due to lack of context, but help the model handle brief inputs. Long sentences, while resource-intensive, provide more context for learning.

The variations in sentence lengths require a flexible span masking strategy. For shorter sentences, we may need to mask fewer words to retain ample context, while for longer ones, we could mask larger spans without considerably diminishing the contextual information. Given our objective of training separate adapters for each language, we will tailor our masking approach accordingly to maximize learning efficacy from both corpora.

Likewise, the word frequency analysis of our corpora offers insights into the patterns of language use. In the English corpus (Figure 7), we found a set of the most frequent words that predominantly include function words and a few content words. A similar pattern

⁵https://cls.corpora.uni-leipzig.de/en/deu_news_2022

⁶<https://wortschatz.uni-leipzig.de/en/download/English>

⁷<https://wortschatz.uni-leipzig.de/en/download/French>

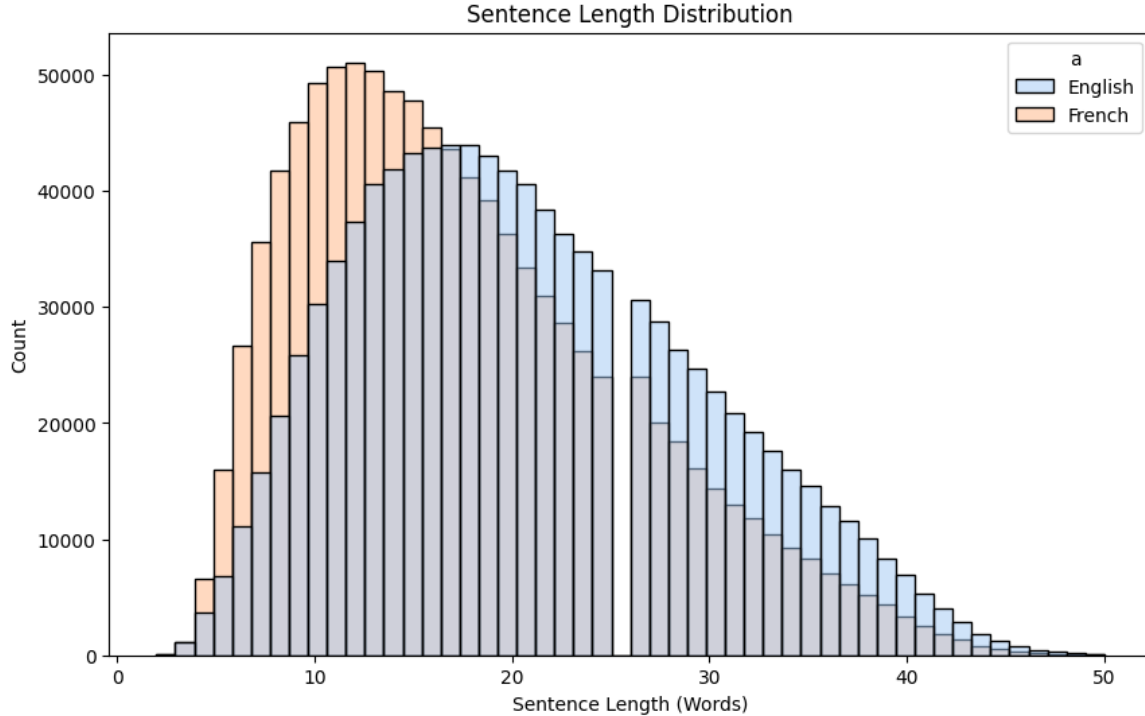


Figure 6: Sentence Length Distribution in English and French Corpora

(Figure 8) is observed in the French corpus. These frequent words, while not carrying significant semantic information, play an important role in the structure of the language.

The analysis of unique words in our corpora indicates the richness of the language vocabularies. As seen on Figure 9 we found around 25,000 unique words in the English corpus, pointing to a substantial diversity in the vocabulary. The French corpus presents even higher diversity with approximately 35,000 unique words. This difference underscores the breadth and variety of language use captured in our corpora.

5.3 Fine-tuning Data for Task Adapter

The fine-tuning dataset is a combination of several English Question/Answer (Q/A) datasets, namely SQuAD 2.0 [76], Eli5 [26], QuAC [19], and NarrativeQA [47]. We decided on combining these datasets because we wanted to include diverse questions types in our fine-tuning process. While SQuAD 2.0 mostly includes ‘What’ questions, Eli5 has a lot of ‘Why’ questions, as such, combing these different datasets provides us with various question types as well as various context length which we predict will make our model

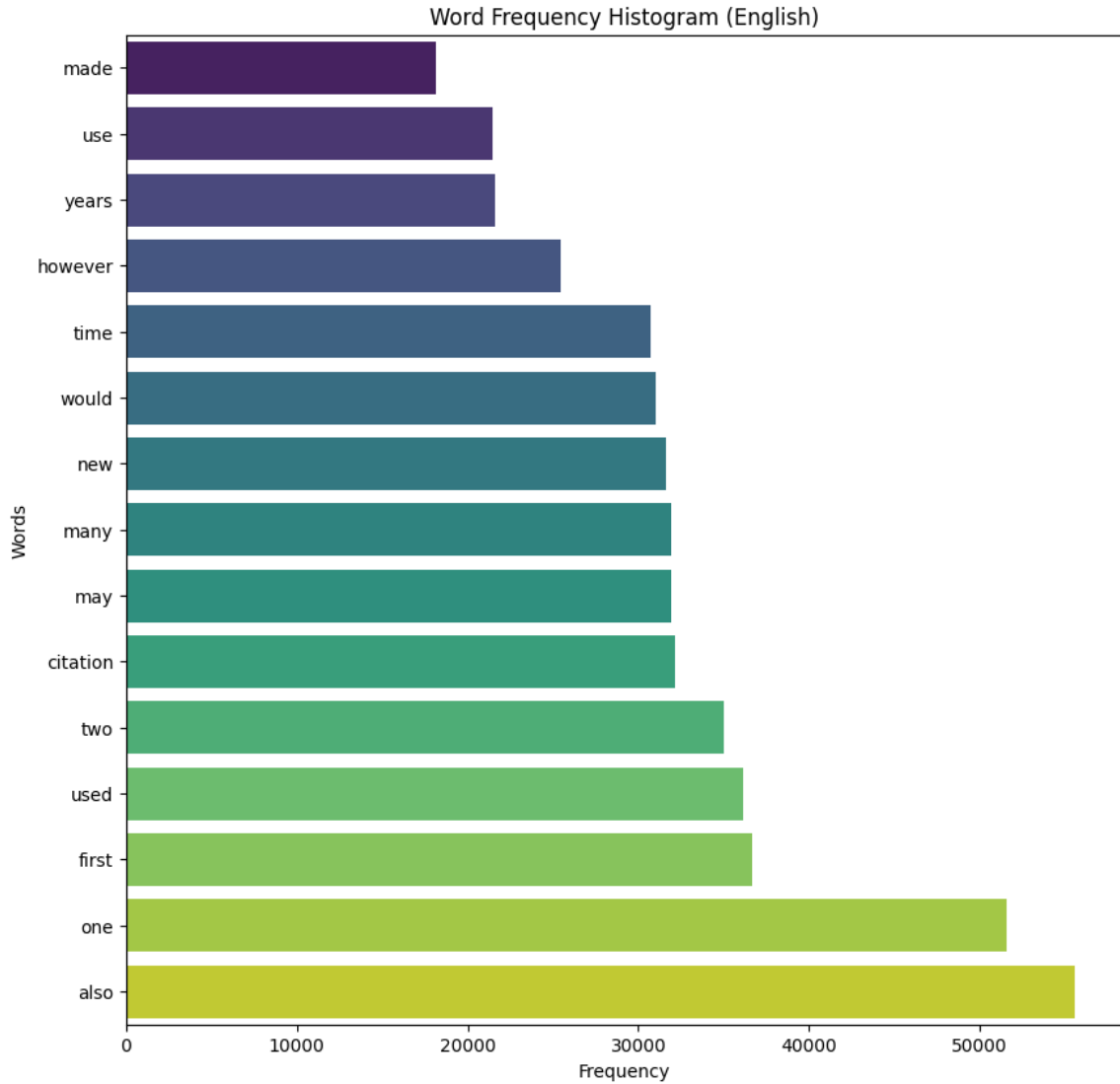


Figure 7: Top 15 words with highest frequency in English corpora

better at producing diverse questions and handling various size context.

SQuAD 2.0: SQuAD 2.0 [76] is a modified version of the SQuAD 1.0 [75] dataset that includes a new layer of difficulty. It includes the original data as well as over 50,000 unanswered questions. Contributors designed these difficult questions to appear similar to answerable ones, increasing the dataset’s difficulty. The fundamental goal of SQuAD is to assess machine learning models’ ability to understand natural language and produce appropriate answers.

The dataset is primarily composed of questions derived from Wikipedia articles. Each

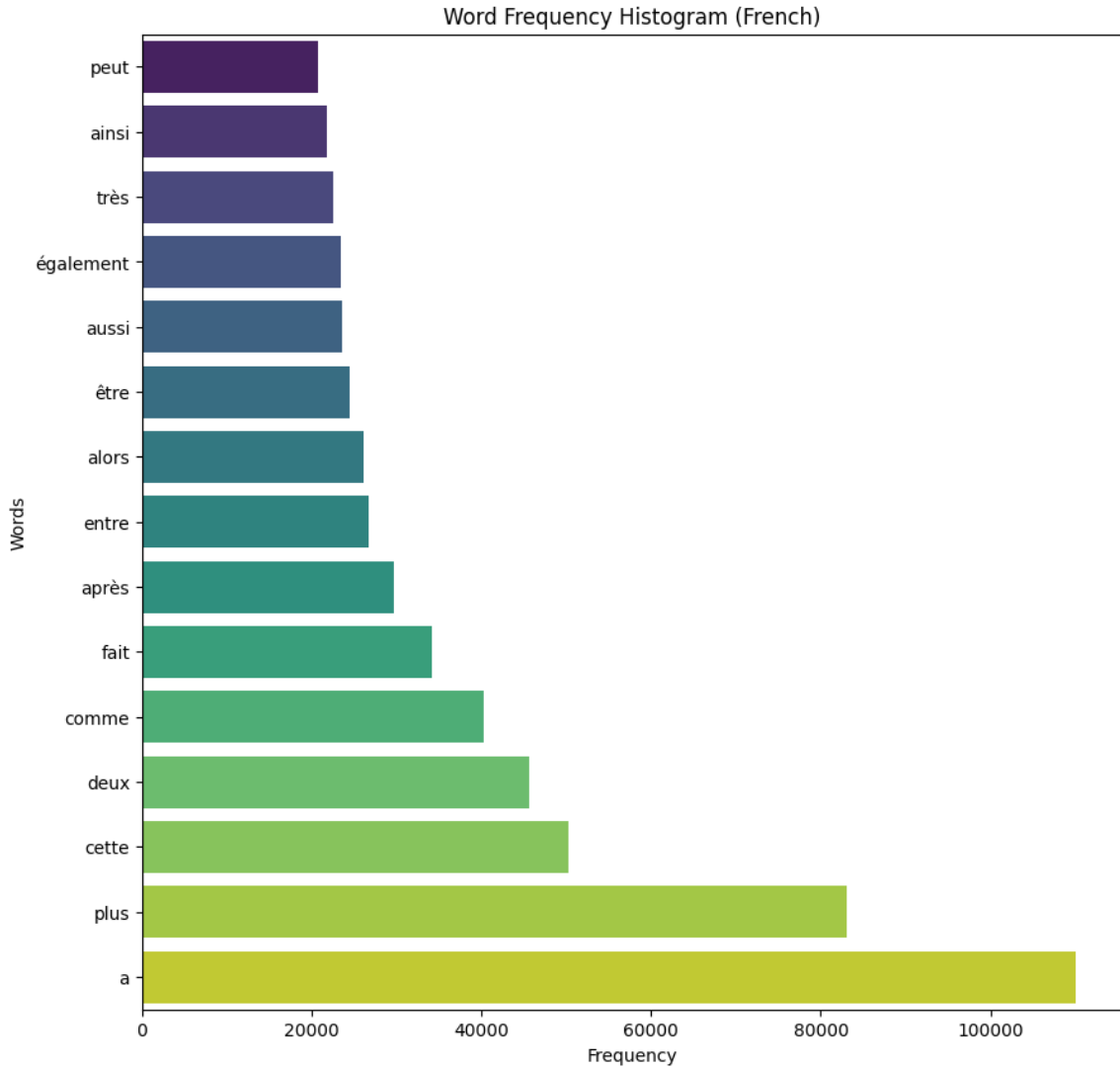


Figure 8: Top 15 words with highest frequency in French corpora

question is accompanied by a relevant paragraph from the corresponding article, which offers context and contains the answer. SQuAD 1.0 [75] contained around 100,000 question-answer pairings. SQuAD 2.0 [76], on the other hand, has approximately 150,000 question-answer pairs, including both answerable and unanswerable questions, offering a more difficult test for machine comprehension.

Eli5: The ELI5 dataset, as introduced by Fan et al. [26], is a product of the Reddit community “Explain Like I’m Five”. It is unique due to its emphasis on long-form question-and-answer sessions, with responses averaging 131 words. The questions, pri-

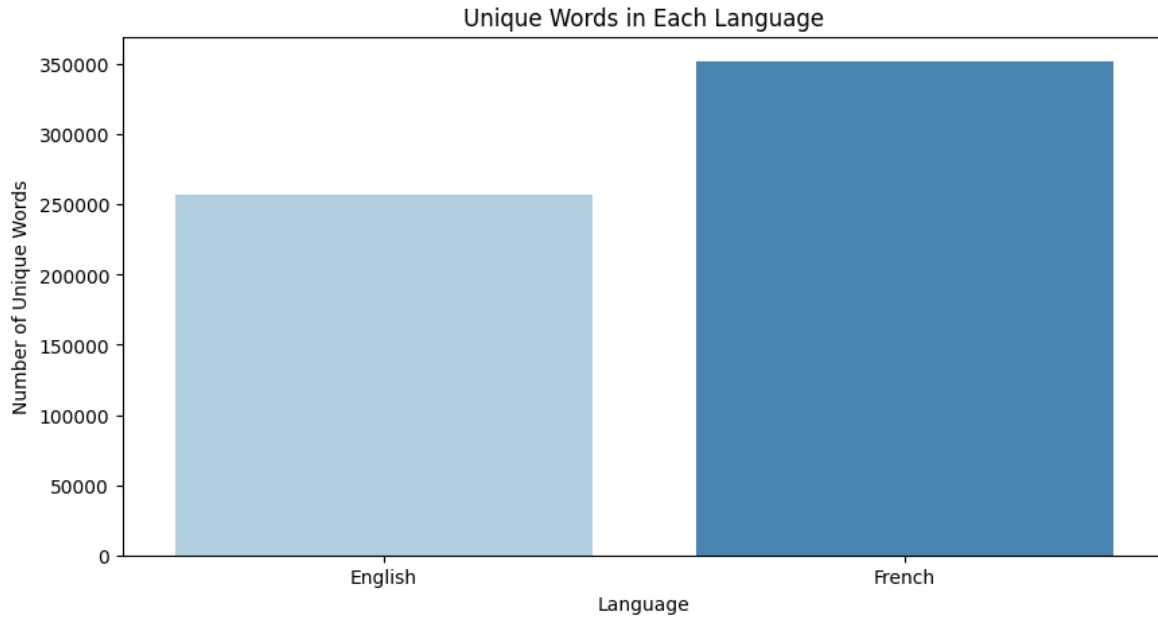


Figure 9: Number of unique words in the English and French corpora

marily of the ‘what’, ‘why’, and ‘how’ variety, require deep, extensive responses, often relying on multiple supporting references.

The main challenges with this dataset involves isolating relevant information from long source documents and generating lengthy, paragraph-level responses to multifaceted, diverse questions. These questions are often so intricate that they cannot be simply addressed with a short response or a keyword from a reference document.

QuAC: Question Answering in Context, or QuAC, is a big dataset that has dialogues pulled from Wikipedia articles. It has about 14,000 of these dialogues that total around 100,000 question and answer pairs. But QuAC is different from datasets like SQuAD 1.0 [75] because in those, the person who asks the question already knows the answer. With QuAC (and also Eli5), they do not. It is designed to look a lot like the types of questions people type into search engines.

What QuAC does is it tries to be like real-life conversations where people are looking for information. One person asks questions about a Wikipedia article that they cannot see, and someone else gives answers based on that article. But this can be tricky. Sometimes the questions don not have one right answer, or they depend on the context, or they could

be answered in many ways. That is why QuAC is really important for people who are studying multi-turn question answering

NarrativeQA: NarrativeQA [47] is a robust dataset consisting of over 46,000 human-authored queries and answers derived from 1,567 book narratives and film scripts. The length and complexity of these sources require a higher level of comprehension and deductive reasoning than traditional news articles or Wikipedia entries.

This dataset, introduced by Kočiský et al. [47], is distinctive in that it challenges users to synthesize information about the narrative’s embedded events and relationships. Mou et al. [63] emphasized in 2021 that a significant number of queries in NarrativeQA focus on narrative events and their interrelationships.

5.3.1 Data Transformation

As previously discussed, we utilize QA datasets to train our AQG systems. This involves pre-processing and merging four different QA datasets to create a dataset consisting of question-context pairs, known as the QG) dataset. We do not include answers in the dataset because our training process does not require them.

These datasets were retrieved using the datasets function in the HuggingFace library. When accessed through HuggingFace, the datasets are presented as DatasetDict objects. Similar to a Pandas DataFrame, a DatasetDict object is designed for machine learning tasks and can handle large datasets that cannot fit into memory.

	No. of contexts	
	Training	Validation
SQuAD 2.0	130,319	11,873
Eli5	272,634	9,812
NarrativeQA	32,747	3,461
QuAC	83,568	7,354

Table 2: The number of training and validation data for fine-tuning QG task adapter

After we acquired the datasets, we extracted the questions and their contexts from

each source, combining them into one dataset. The resulting question-context pairs were then stored in a pickle file. After the preprocessing and merging, we ended up with about 519,000 pairs of question-context pairs for our QG dataset. Table 2 shows the number of training and validation data we merged from each of the datasets.

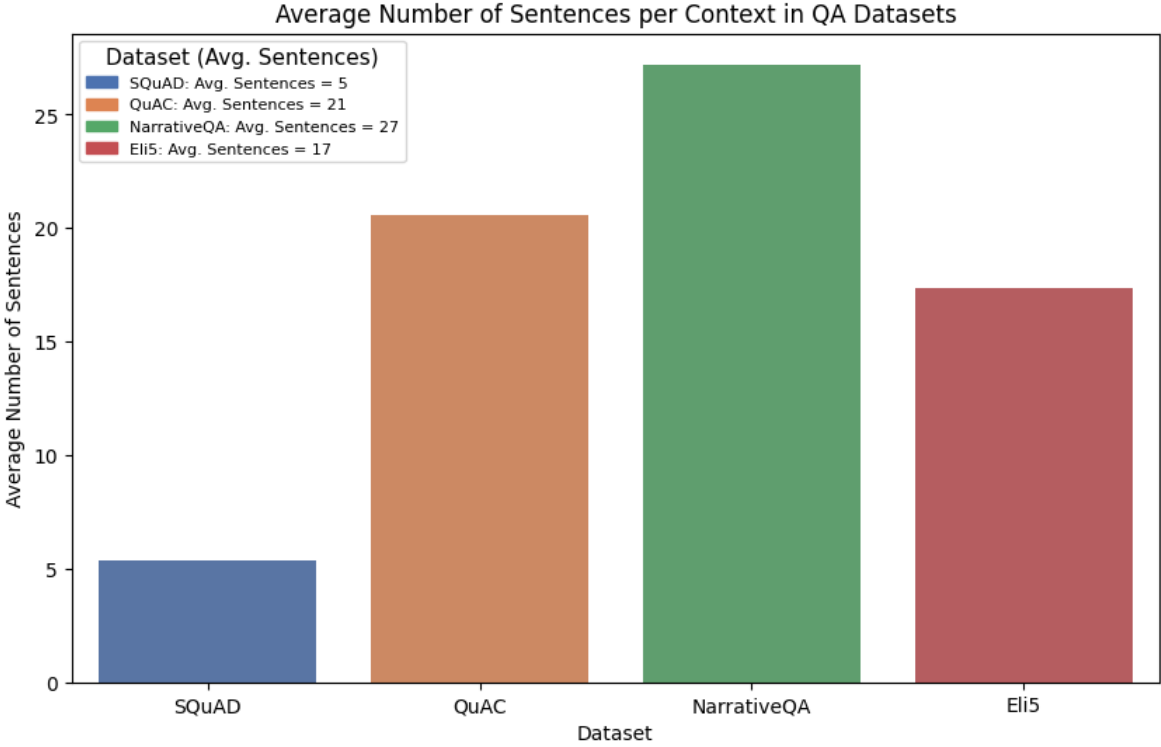


Figure 10: QG dataset distribution

Refer to Figure 10 to see the average number of sentences per context from this dataset. We find that on average, the SQuAD dataset provides the shortest contexts, at about 5 sentences, while the NarrativeQA dataset offers the longest, averaging at 27 sentences. This variety is useful for our fine-tuning task, as it introduces contexts of different lengths, assisting the model in generating questions.

6 Methodology

Our study is aimed at creating a system capable of generating relevant questions from a given text excerpt or context. This aim is to design a system capable of accommodating multiple languages. However, a significant challenge arises due to the scarcity of multilingual question generation datasets. Translating the available English Question Generation (QG) datasets to other languages for the purpose of training a multilingual model was considered. Still, this strategy shifts the focus from question generation to translation, undermining the original goal of our study.

Additionally, fine-tuning LLMs on extensive datasets poses another problem. It is a process that is computationally expensive and time-consuming. Given our goal of designing an architecture that is not computationally expensive, the collection of translated multilingual QG datasets becomes an unfeasible option.

To address these challenges, we experimented with two AQG systems. The first main proposed pipeline incorporates the use of adapters [40] with a multilingual pre-trained language model. The second method, makes use of generative pre-trained LLMs and the Langchain framework⁸ to create a simple pipeline based on prompting.

Section 6.1 provides a detailed explanation of our adapter based system. and Section 6.2 provides details on the second pipeline.

6.1 Adapter-Based System

The architecture for our adapter based system is inspired from the MAD-X framework [69]. Adapters are small neural layers incorporated into PLMs. Their function is to facilitate task-specific training, fine-tuning only the parameters within the adapter layers, while the original parameters of the pre-trained model remain untouched. This promotes efficiency as, instead of needing to store entirely pre-trained models for each task, only the relatively small adapter layers need to be stored. Consequently, these adapters enable the sharing and reuse of models across different tasks in a more resource-effective manner.

⁸<https://www.langchain.com>

6.1.1 Overview

Model Selection In building our adapter based system, the first step was to pick the right PLM that would cohesively integrate with our proposed architecture. As referenced in section 3.4, there has been a notable growth in PLMs in the NLP sector. With our goal centered on language generation, it became essential to explore PLMs that excel in this area, especially those adept at multiple languages and have been open-sourced. Among the prominent models like mBART [56], XLM-RoBERTa [20], and GPT-2 [73], we decided on Google’s mT5 model [95], designed specifically for its capability in generating text across various languages in a human-like manner [1, 94, 28].

This model is of the T5 (Text-to-Text Transfer Transformer) family, designed for various language tasks. At its core, mT5 uses an encoder-decoder architecture. This means it first “encodes” or reads the input text, and then “decodes” or generates a response. What makes mT5 special is its ability to handle multiple languages, making it versatile for different tasks. For natural language generation, mT5 is a good option because its design allows it to understand context and produce coherent and relevant text. Its foundation on the T5 framework ensures it benefits from proven techniques and research, making it both reliable and efficient for generating human-like text.

System Architecture Drawing inspiration from the MAD-X framework [69], we experiment with two types of adapters: Language Adapters and Task Adapters. The key principle underlying these adapters is that the modular adapter framework allows us to interchange different language adapters during inference, facilitating cross-lingual transfer. This capability is especially useful as the task adapter is exclusively fine-tuned on an English dataset due to data constraints in other languages.

To this effect, we take inspiration from this framework and design a similar architecture where the multilinguality of our system is facilitated by the language adapters while the question generation task is handled by the task adapter. Figure 11 shows the MAD-X framework inside a Transformer model.

The specifics of training these adapters and their role in the multilingual question generation system are further elaborated upon in the following sections.

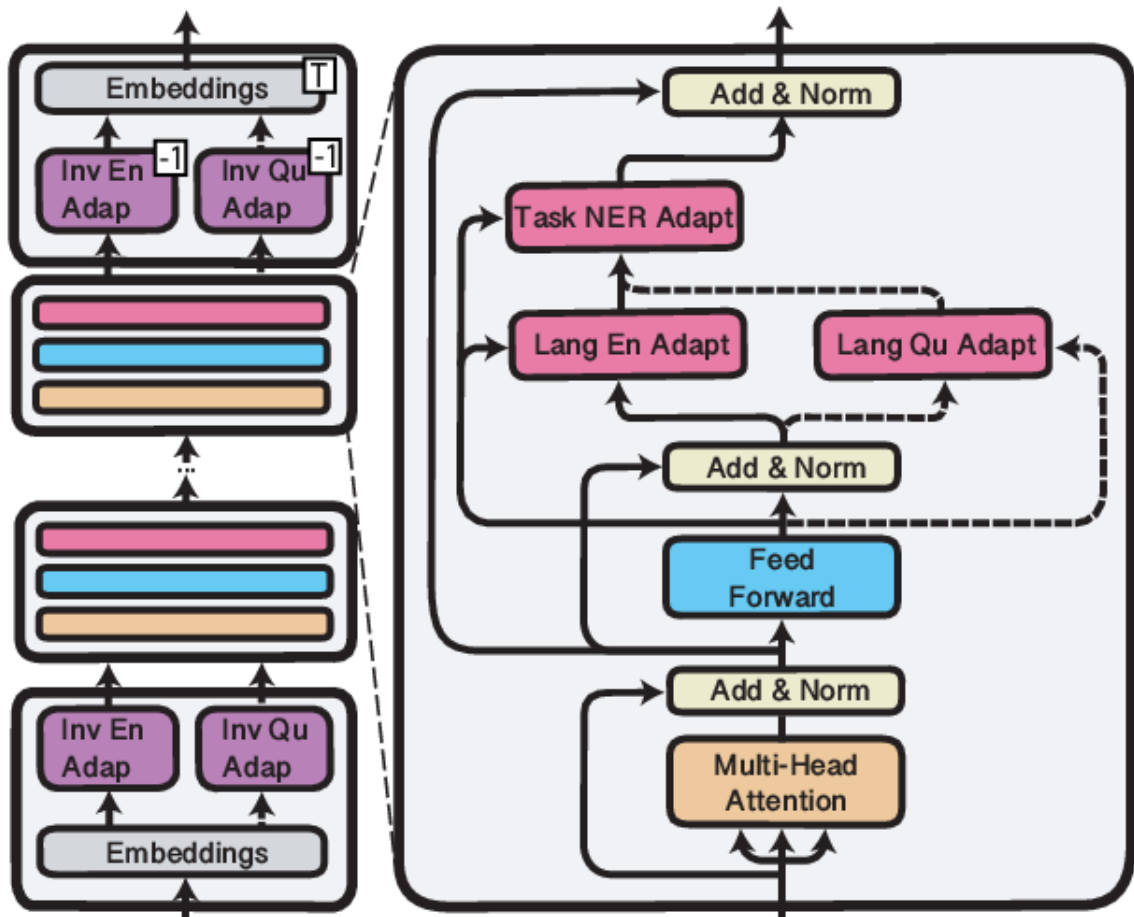


Figure 11: The MAD-X structure within a Transformer model takes input embeddings into an invertible adapter, which then reverses into the tied output embeddings. Every Transformer layer incorporates language and task adapters. Language and invertible adapters learn through masked language modeling (MLM) while the existing multilingual model remains unchanged. For specific tasks, task-dedicated adapters overlay the primary language adapters (solid lines). During zero-shot cross-lingual shifts, primary language adapters are substituted with target language ones (dotted lines) [69]

6.1.2 Adding adapters to mT5

AdapterHub⁹, serves as a streamlined platform for integrating, training, and applying adapters, especially for Transformer-based language models. At its core, AdapterHub offers two main components: the adapter-transformers, which enhance the Huggingface’s

⁹<https://adapterhub.ml>

Transformers library with adapter functionalities, and The Hub, a specialized repository containing pre-trained adapter modules.

At the moment, AdapterHub does not natively support the mT5 model. Therefore, our initial step was to modify the mT5 model to ensure its smooth integration with the adapter-transformers setup.

Incorporating adapters into the mT5 architecture was a direct process, guided by the comprehensive resources available in the AdapterHub repository. The task was not just about attaching adapters into the mT5 but involved a careful modification of the architecture to ensure a seamless integration. Here is a distilled overview of the steps taken:

1. **Model Forward Pass Logic Modification:** A significant chunk of the integration required tweaking parts of the model’s forward pass logic. Instead of altering the entire model, changes were made strategically within the model’s class to accommodate the adapter layers.
2. **Utilizing Adapter Mixins:** Essential mixin components were added to the Transformer module classes. These mixins, provided by the AdapterHub, come equipped with the necessary functionality to embed the adapter layers effectively within the architecture.
3. **Model and Config Adjustments:** The base model was supplemented with specific mixins to introduce the adapter layers. We also ensured the proper execution sequence for operations, such as the adapter layer forward pass, by referring to examples of other models (T5) that had undergone similar modifications.
4. **Config Class Refinement:** The configuration class was tweaked to align with the adapter requirements. This primarily revolved around ensuring the presence of specific attributes vital for the adapter’s functioning.
5. **Establishing the AdapterModel Class:** As a final step to our integration process, an adapter-supporting model class was introduced. This class is instrumental

in adding versatility to the architecture, allowing the efficient addition and management of multiple prediction heads.

Through this methodical process, rooted in the best practices outlined by AdapterHub’s documentation, we achieved a seamless and functional integration of adapters into the mT5 architecture.

6.1.3 Language Adapters

After modifying the mT5 model to include adapters, we focused on training the two language adapters for our system: English and French. As referenced in Section 5.2, the Wikipedia corpus served as our primary dataset.

The language adapters were trained using the span Masked Language Modeling objective, which enables the adapter to effectively grasp and learn the structure of a language. Within this method, specific text spans from the corpus are masked using tokens like ‘`extra_id_0`’. Following this, the model is trained to reproduce the masked spans. This technique is consistent with the pre-training method used in the mT5 model. As a result, we prefer this methodology over traditional Masked Language Modeling techniques.

For this task, the adapter method implemented is the *PfeifferInvConfig* configuration. This was proposed by Pfeiffer et al. [69] and is tailored to learn language-specific transformations as shown in Figure 11. It is particularly efficient for zero-shot cross-lingual transfers.

To illustrate, Table 3 offers a sample of how text input and output appear during the span MLM technique. Essentially, the training objective mirrors a fill-in-the-blanks task. It is an expanded version of the continuation task, as the “blanks” can also appear at the end of the text.

Figure 12 provides a detailed overview of the language adapter training pipeline. This starts with pre-processing, where masks are added to the input text spans. Following this, the adapter is integrated with the mT5-base model. During this phase, the model’s parameters remain static, but the adapter’s parameters are active. The subsequent training targets only the active parameters using the span MLM technique. Once the training concludes, only the trained adapter is retained.

Original	LCT students are excited for the annual meeting.
Inputs	LCT $\langle X_i \rangle$ excited for the $\langle Y \rangle$ meeting.
Targets	$\langle X \rangle = \text{students are}$, $\langle Y \rangle = \text{annual}$, $\langle Z_i \rangle = \text{end token}$

Table 3: Example of a sentence during span masked language modeling. The target sequence consists of the dropped-out spans, delimited by the same sentinel tokens used in the input plus a final sentinel token $\langle Z \rangle$ to mark the end of the target sequence. [95]

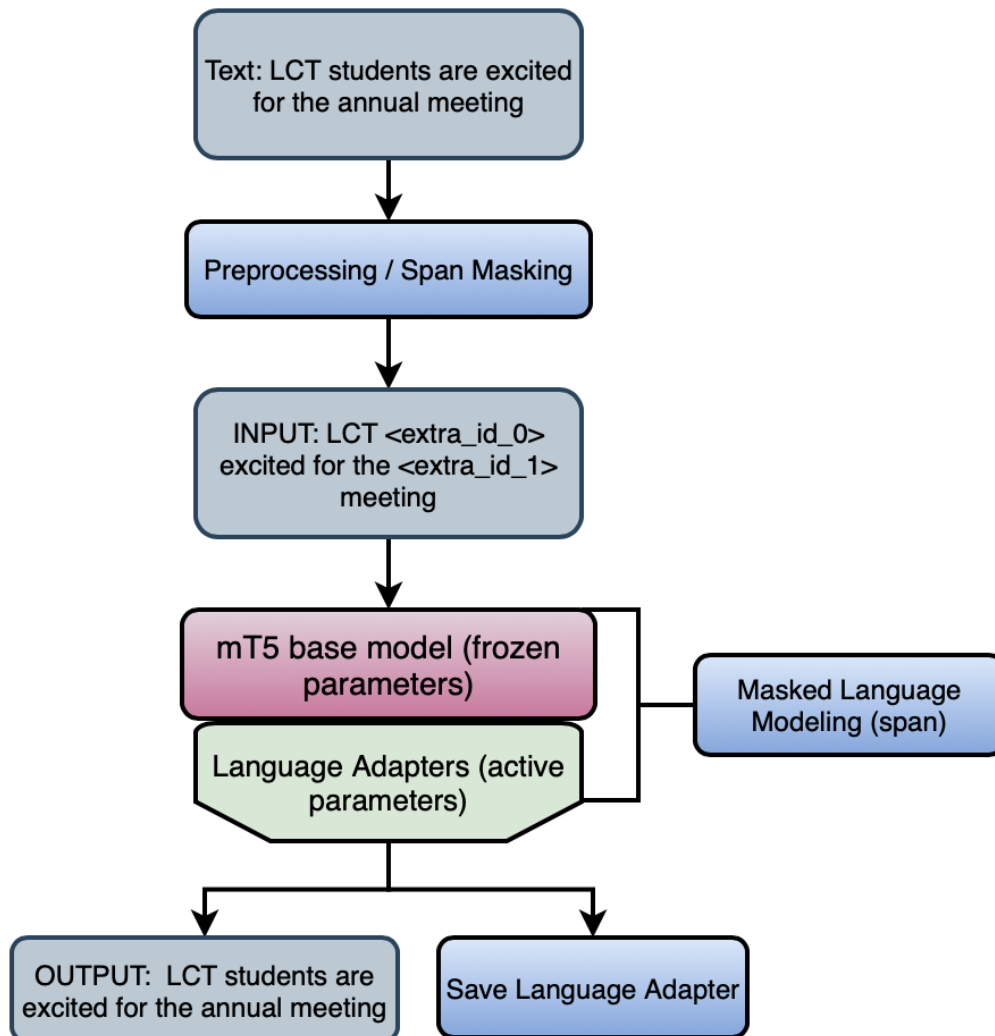


Figure 12: Overview of training language adapter pipeline

6.1.4 Task Adapter

After training the language adapters, our next step was to fine-tune the model specifically for question generation. We utilized the source language adapter and integrated the task adapter to fine-tune it for this specific task.

The Task Adapter is trained using our prepared QG dataset, as detailed in section 5.3, where the context is treated as input and the corresponding questions act as targets. For fine-tuning, we employ the Stack method from AdapterHub. This allows us to place the source language adapter, in our instance the English Language Adapter, ahead of the task adapter. This strategy enhances training for our specific downstream question generation task. Our chosen method for this process is the *PfeifferConfig* adapter, which inserts an adapter layer after the feed-forward block in each transformer layer.

For clarity, Table 4 demonstrates an example of input and target text during the training phase. It is important to note the distinction from models like T5 [74]. Unlike T5, mT5 [95] was not trained on supervised downstream tasks. As it was solely pre-trained in an unsupervised manner on mC4 (multilingual colossal clean Common Crawl corpus), there is no added benefit of incorporating a task prefix when fine-tuning for a singular task. Therefore, in our fine-tuning process, we have opted against using prefix tokens.

Figure 13 illustrates the task adapter training pipeline in detail. The process starts with contexts used as the input text. The system setup involves beginning with the mT5-base model and then adding the source language adapter on top. After this, we introduce a new adapter layer. In this configuration, the mT5-base model and the source language adapter parameters are kept constant. The new adapter layer, set to become the QG task adapter after some fine-tuning, is the one we adjust. Once training is complete, only the task adapter is saved.

6.1.5 Final Pipeline

During the inference stage, both adapters are combined with the base model, enabling question generation. The multi-lingual proficiency of our system is primarily attributed to the multilingual base model complemented by the pre-trained Language Adapters.

Input_text (Context)	Kublai Khan did not conquer the Song dynasty in South China until 1279, so Tibet was a component of the early Mongol Empire before it was combined into one of its descendant empires with the whole of China under the Yuan dynasty (1271–1368). Van Praag writes that this conquest "marked the end of independent China," which was then incorporated into the Yuan dynasty that ruled China, Tibet, Mongolia, Korea, parts of Siberia and Upper Burma. Morris Rossabi, a professor of Asian history at Queens College, City University of New York, writes that "Khubilai wished to be perceived both as the legitimate Khan of Khans of the Mongols and as the Emperor of China. Though he had, by the early 1260s, become closely identified with China, he still, for a time, claimed universal rule", and yet "despite his successes in China and Korea, Khublai was unable to have himself accepted as the Great Khan".
Target_text (Question)	Which dynasty ruled all of china?

Table 4: Example of the input and target text passed on to the QG Task adapter during training on downstream task

The final pipeline of our system is shown in Figure 14. Initially, client data undergoes pre-processing: documents are segmented into contexts and stored in a Knowledge Base

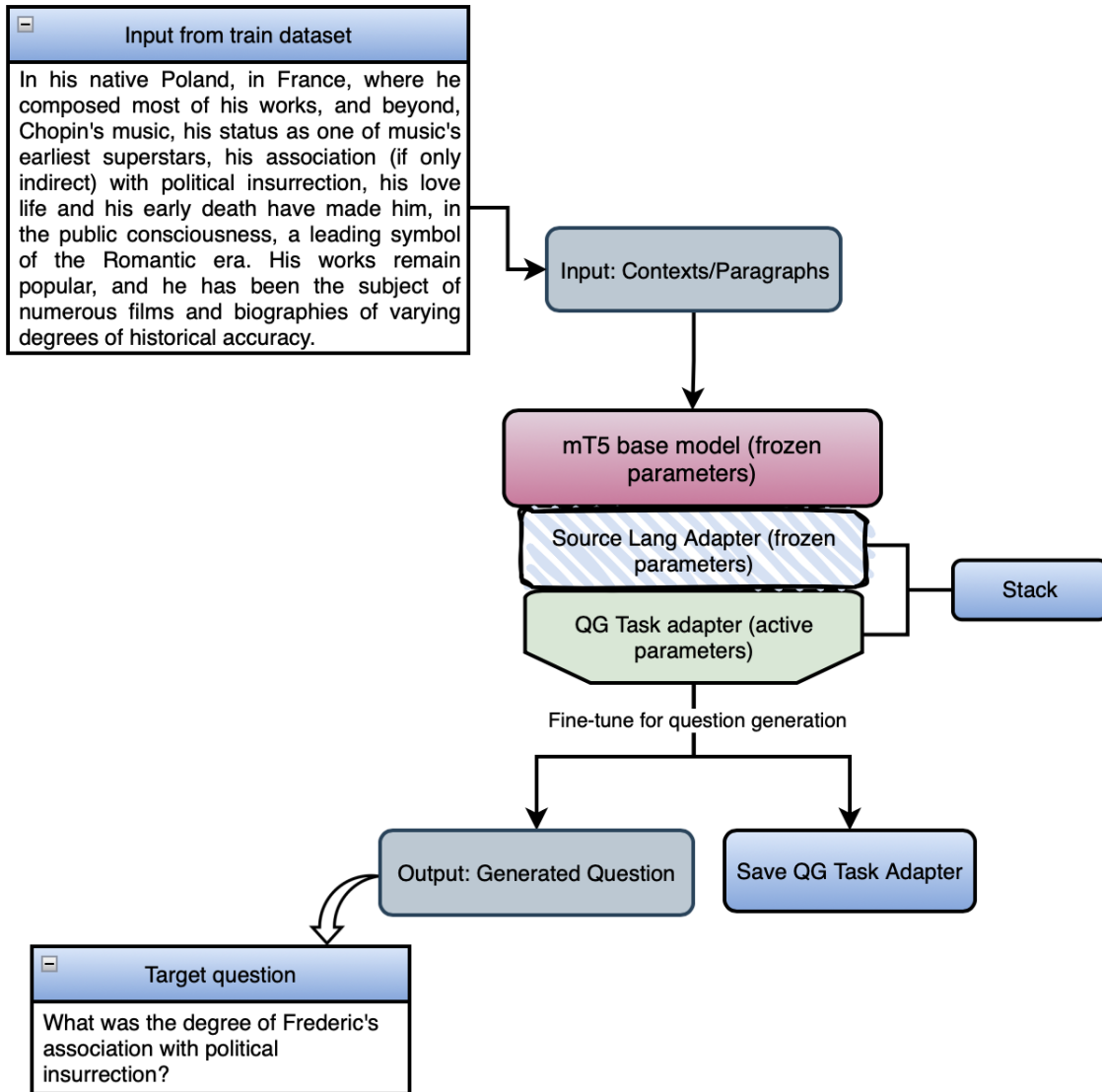


Figure 13: Overview of training Question Generation Task adapter pipeline

(KB). For each specific client chatbot, the system identifies the language of the text within the KB. Once the language is determined, the appropriate pre-trained language adapter is selected and layered onto the base model. Following this, the QG task adapter is incorporated using the AdapterHub Stack method. The text, or context, is then input into the system, enabling the generation of the desired number of questions for each context. Once questions are generated they are stored back into the KB along with their corresponding contexts. In Figure 14, you can observe an example of actual client text passed to the pipeline and the two questions that were generated in the end.

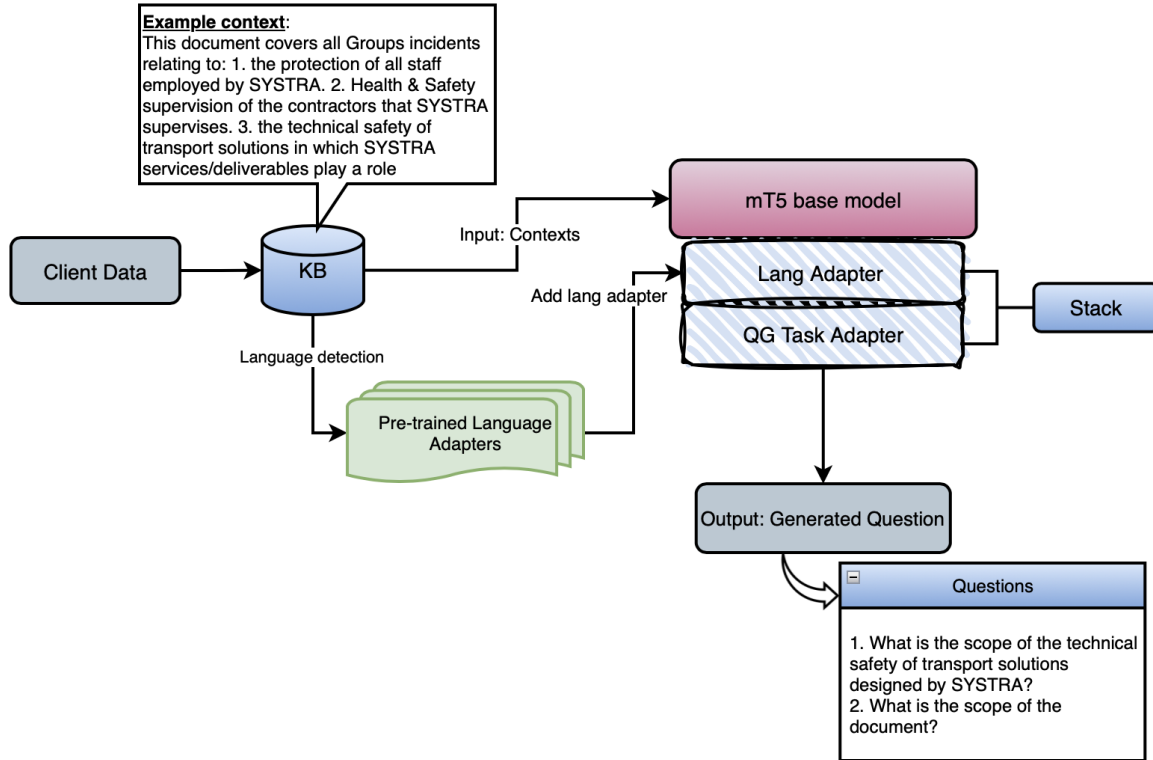


Figure 14: Overview of final pipeline during inference

6.1.6 Hardware and Hyperparameters

Hyperparameters We decided against in-depth hyperparameter tuning due to resource and time constraints. Instead, we performed manual adjustments across few iterations, always closely monitoring the training. Our experiments involved different learning rates, batch sizes, and number of epochs. Additionally, we used a linear learning rate scheduler and introduced weight decay. To ensure the best performance and avoid overfitting, we applied an early stopping strategy, preserving the model when it achieved its best results. For the training, we relied on the AdapterTrainer class from the Huggingface API Trainer.

Hardware For the training phase of the adapters, an A100 Nvidia GPU, accessible via Google Colab, was employed. The inference stage utilized a T4 GPU, also courtesy of Google Colab. Preliminary testing, debugging, and dataset compilation were conducted on a V100 GPU within the same platform.

6.2 LLM Langchain System

Our LLM langchain system is built upon the foundation of current generative models, notably the OpenAI GPT model series [73, 13]. Instead of the resource-intensive approach of training or fine-tuning large pre-trained language models (PLMs) like mT5 [95], we opt for a prompting-based method. Given the recent advancements in generative models, as detailed in section 3.4, our second pipeline focuses purely on prompting without any additional training or fine-tuning. This approach not only eliminates the need for GPU clusters and extensive training time but also mitigates the challenge of requiring training data. The effectiveness of this strategy is grounded in the fact that these LLMs come pre-trained with vast datasets and contain billions and trillions of parameters.

6.2.1 Model Selection

When designing our LLM langchain system, we needed to choose the best model for prompt-based interactions. As discussed in Section 3.4, models like GPT-2 and GPT-3 from OpenAI have shown great ability in understanding and producing human-like text. A recent advancement on this front is the GPT-3.5 Turbo which powers ChatGPT. This model has been extensively trained on conversational data, enabling it to effectively understand and respond to natural language queries. There are several versions of the GPT-3.5 Turbo¹⁰ available via the OpenAI API service, with varying token capacities, from 4k to 16k. For our LLM langchain system, we selected the GPT-3.5 Turbo API from OpenAI. The cost for using this API is \$0.002 per 1K tokens.

6.2.2 System Architecture

The architecture of our system is based on the Langchain framework, tailored for working efficiently with LLMs (Large Language Models). LLMs are versatile, meaning they can handle a multitude of tasks. However, when it comes to in-depth, specialized knowledge, particularly in niche areas like psychology or legal matters, they might not always deliver precise answers.

¹⁰<https://platform.openai.com/docs/models/gpt-3-5>

LangChain addresses this limitation by employing a unique method: text is divided into segments or concise summaries, which are then embedded within a vector framework. When a query is posed, the system scans these segments for resemblances to provide the most relevant response.

Our system is specifically built using the Langchain framework in conjunction with the OpenAI 3.5-turbo API. We focused our experiments on two distinct prompt + context scenarios. The architecture is straightforward: it accepts a prompt accompanied by a context of varying length. The GPT-3.5 Turbo processes the prompt and the given context to produce an answer. Notably, our system does not require any training or fine-tuning. Hence, we bypass the training datasets discussed in Chapter 5, instead sourcing client data (contexts) directly from the knowledge base mentioned in Section 2. The techniques we employ for prompting, and how we use context from the knowledge base, will be elaborated further in the subsequent sections.

Prompt + Context In the first scenario, our approach mirrors the intuition adopted in the adapter-based pipeline: generating questions from paragraph contexts. We create a prompt instructing the model to derive relevant questions from the provided context (paragraph). The API then processes this for question formulation. More specifics on the prompt employed can be found in Appendix A.1

Prompt + Title For our second scenario, we shift our focus to using document titles rather than paragraph contexts as the basis for question generation. These titles are essentially the headings found in client documents, parsed during the pre-processing phase, as outlined in Section 2. We posit that prompting the model with just the headings/titles, instead of full-length paragraphs, might yield questions that are more contextually relevant. This approach not only aims at increasing the relevancy but also conserves tokens per request, leading to a more cost-effective solution. The specific prompt utilized in this context is detailed in Appendix A.2.

Figure 15 provides an overview of the LLM Langchain system architecture. This setup works with any LLM that has an API. The core component, LLMChain function, includes both a PromptTemplate and a language model, which can be an LLM or a chat model.

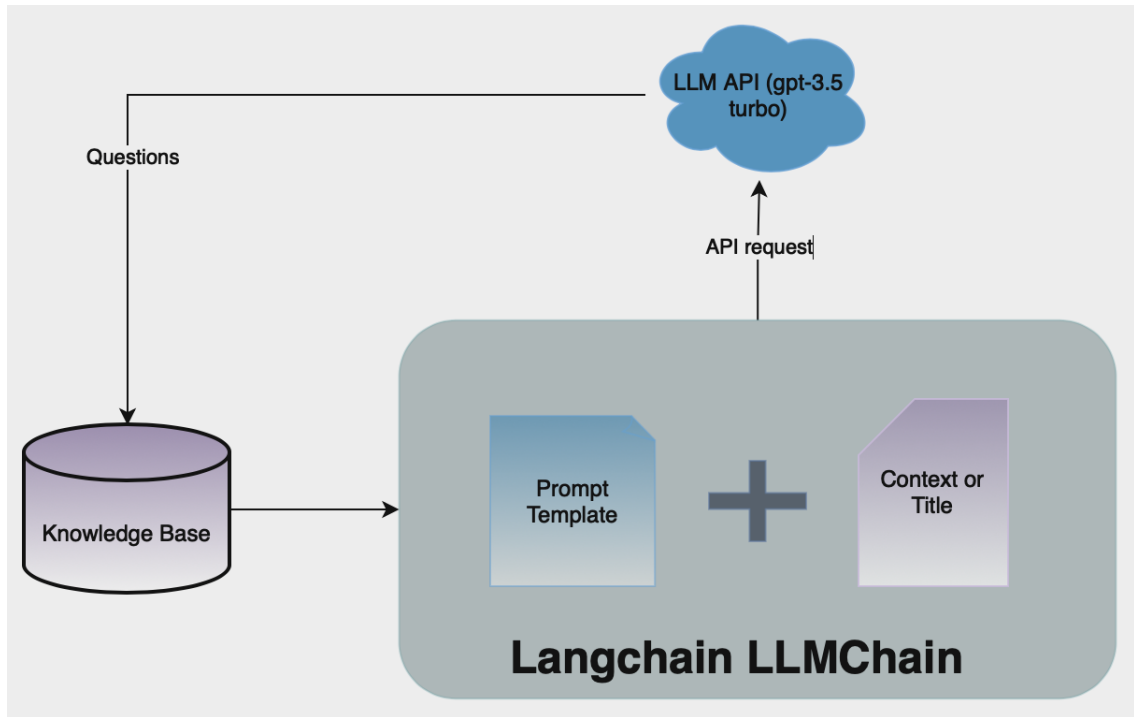


Figure 15: Overview of LLM langchain pipeline

It takes the given input, formats it using the prompt template, and sends it to the LLM. Then, it collects the response from the LLM. The figure illustrates the pipeline for both of our methods, whether using context or title. All these contexts and titles are saved in our knowledge base, which can be accessed using SQL scripts. Once the LLM creates a question, it is stored in the knowledge base with its related context or title.

7 Evaluation and Results

Evaluating the Automatic Question Generation (AQG) task presents unique challenges. In this project, our primary objective is to determine if adding the generated questions to the knowledge base enhances the DearBot solution’s ability to deliver accurate responses to user queries. Our evaluation process is twofold. Firstly, we use automatic evaluations to measure the similarity between our systems’ generated questions and their associated contexts, thereby assessing the quality of the questions produced. Secondly, we utilize the Testbook evaluation system from Synapse Développement, which features question-answer pairs, to evaluate the chatbot’s performance when the generated questions are integrated into the knowledge base.

7.1 Baseline Models

To gauge the performance of our two systems, we established two straightforward baselines. The first baseline involves fine-tuning an mT5-base model on the English QG dataset (Section 5.3) without any prior pre-training. This model underwent fine-tuning using the same hardware configurations as the Adapter-based approach. Additionally, our second baseline for comparison is the question generation model currently used by Synapse Développement. This model is both rule-based and template-based, generating questions from document headings or titles using templates.

7.2 Generated Question evaluation

We have chosen the BERTScore [99] and OpenAI similarity score to measure how closely our generated questions relate to their respective contexts. BERTScore computes the cosine similarity between the BERT embeddings of two texts, offering a more granular perspective of semantic similarity compared to traditional metrics. The OpenAI similarity score is the same but in this case we use the ‘text-embedding-ada-002’ model¹¹ to get the embedding of the texts. The point of adding OpenAI similarity score is to see if there could be any differences in results between the two embedding similarity scores. While

¹¹[OpenAI text embedding api](#)

traditional metrics like BLEU and ROUGE are useful for many NLP challenges, they are not always ideal for assessing generated text. This is because they focus on n-gram overlaps and can miss the semantic accuracy needed for question generation. In contrast, BERTScore and OpenAI similarity score consider the context of each word, not just n-gram matches, making them more suitable for tasks where the meaning of the text is important.

7.2.1 Data

For the automatic evaluation of the similarity between generated questions and context, we utilize two client data sets provided by Synapse Développement: one in English and the other in French. These contexts, sourced from various document formats like html and pdf, are accessible through the Knowledge Base (KB). While the length of these contexts in the KB varies, their structure, for the most part, mirrors that of the datasets we used during our adapter model training. We use these datasets to obtain the BertScore and OpenAI similarity scores for both systems we developed, as well as the two baseline models. Detailed statistics about this dataset are available in Table 5.

	No. of contexts	Avg. # of sentences
English	843	23
French	378	18

Table 5: Details of the evaluation dataset

7.2.2 Results

Table 6 displays the BertScore and OpenAI similarity scores of the models based on the questions they generated for French contexts. The GPT-3.5 turbo (context) model performs the best, achieving a BertScore of 76%. This surpasses the GPT-3.5 turbo (title)¹² model by 8%, the mT-5 and synapse baseline by 21%, and the Adapter model by 14%. For the OpenAI similarity scores, the GPT model achieved an impressive 94%. The

¹²Note that while the questions in this pipeline is generated based on titles, we make use of the corresponding passage of the titles (i.e contexts) to compute the similarity scores

mT-5 and the older Synapse baseline achieved scores of 82% and 79% respectively. The GPT-3.5 turbo (title) model scored 90%, while the Adapter model got 88% surpassing the baselines but not the GPT-3.5 models. These results further highlight the strength of LLMs like GPT-3.5 turbo that have been trained extensively on enormous datasets. Also, it further confirms the advantage of adapter tuning approach over the baselines, proving its consistent performance not just in English, but also in French contexts.

	Question - Context (English)	
	BertScore	OpenAI Score
Baseline (mT5-base)	55%	82%
Baseline (Synapse)	55%	79%
Adapter-Based Model	62%	88%
GPT-3.5 turbo (context)	76%	94%
GPT-3.5 turbo (title)	68%	90%

Table 6: Results for the similarity between English questions and contexts: A comparison of questions generated from baseline models, the Adapter-based system, and GPT systems.

In comparison, the Adapter model, while not surpassing the GPT-3.5 turbo models, does perform better than the baseline models. Its BertScore is 62%, a 7% improvement over the baselines. These results suggest that adapter tuning could be as effective, if not more so, than comprehensive fine-tuning techniques and template-based methods. As for the OpenAI similarity score, the GPT-3.5 turbo (context) model scored 94%. This is notably higher than the mT-5 baseline’s 82%, the Synapse baseline’s 79%, and the Adapter model’s 88%.

Similarly, Table 7 presents the BertScore and OpenAI similarity scores for these models, with respect to the questions they generated aligned to the French contexts. Here, the GPT-3.5 turbo (context) model once again showcased a better performance. It achieved a BertScore of 70%, outperforming the GPT-3.5 turbo (title) model by 7%, the mT-5 baseline by 27%, the Synapse baseline by 21% and the Adapter model by 16%. In terms of OpenAI similarity score, the GPT-3.5 turbo (context) model achieved a high 93% ,

while the mT-5 and Synapse baselines scored 67% and 72% respectively. The Adapter model has a score of 76% outperforming the baselines but still lacking behind the GPT models. These results, further highlight the strength of LLMs like GPT-3.5 turbo that have been trained extensively on enormous datasets. Also, it further confirms the advantage of adapter tuning approach over the baselines, proving its consistent performance not just in English, but also in French contexts.

	Question - Context (French)	
	BertScore	OpenAI Score
Baseline (mT5-base)	43%	67%
Baseline (Synapse)	49%	72%
Adapter-Based Model	54%	76%
GPT-3.5 turbo (context)	70%	93%
GPT-3.5 turbo (title)	63%	86%

Table 7: Results for the similarity between French questions and contexts: A comparison of questions generated from baseline models, the Adapter-based system, and GPT systems.

From the results in both English and French, it is clear that the GPT based models do well. They consistently gets higher BertScores and OpenAI similarity scores. This means the questions they generate fits well with the contexts. These scores show how relevant the questions are to their source. The GPT model’s good scores in both languages mean it understands the context and formulates good questions. However, it is worth noting that the Adapter method, even if it is not as good as the GPT model, is still better than the version Synapse is using now and the mT-5 baseline.

7.3 Evaluation through chatbot performance

As previously stated, we use the Testbook evaluation system from Synapse Développement to assess the performance of our chatbot. In order to understand this evaluation system it is important to first understand the how the QA system of the chatbot at Synapse which plays an important role in this evaluation system. The chatbot answers users in two main

ways: through an extractive method and a generative method.

In the extractive method, when a user asks something, the system looks for a matching question in its knowledge base (KB). If it finds one, it gives the linked answer. On the other side, the generative method also searches the KB for matching questions. But instead of just giving the linked answer, it uses this answer as context to generate a new response.

Based on these methods, our evaluation metrics vary. For the extractive system, we measure the percentage of exact matches between the chatbot’s responses and the reference answers in the testbook. When evaluating with the generative system, we use BertScore and OpenAI similarity scores to compare the generated answers with those in the testbook. In both scenarios, our ultimate objective is to determine the accuracy of questions generated by our adapter-based and GPT models, especially when added to the KB with their respective contexts and titles, in matching user queries with correct answers.”

To execute the evaluation, we first populate the chatbot’s knowledge base with questions generated by the model being tested. Then, the testbook is uploaded to the system. Upon activating the ”Exécuteur tous les tests” button, the system automatically queries the chatbot with each question from the testbook. As the system sequentially processes each query an ’OK’ label signifies an exact match, ’KO’ indicates a mismatch and ’Partial’ is used when either the link to the relevant document or the answer aligns, but not both.

For a clearer visual representation of this evaluation system interface, see Figure 16.

7.3.1 Data

The data used for this evaluation is a testbook comprising question-answer pairs. These datasets have been curated by experts to ensure that the questions align closely with those a person might ask a chatbot. They are comparatively smaller than the ones used for automatic evaluation, with only 60 question answer pairs for the English testbook set and 32 for the French testbook set.

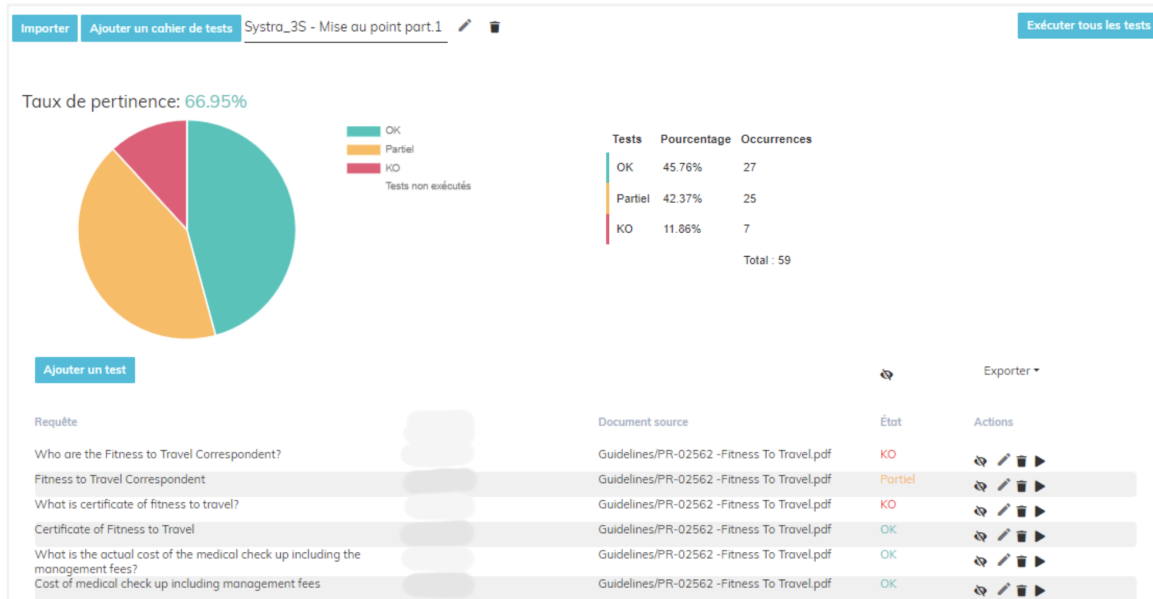


Figure 16: Example of testbook evaluation interface. Here, about 46% of the questions were answered correctly by the bot

7.3.2 Results

Table 8 displays the evaluation outcomes from the English textbook. It is important to clarify again that these scores do not evaluate the questions themselves. Instead, they measure the performance of the chatbot, aiming to understand how the generated questions can enhance the chatbot’s performance.

	English Testbook eval		
	Extractive QA	Generative QA	
	Exact Match	BertScore	OpenAI Score
Baseline (mT5-base)	53%	55%	78%
Baseline (Synapse)	88%	80%	94%
Adapter-Based Model	62%	65%	88%
GPT-3.5 turbo (context)	63%	67%	89%
GPT-3.5 turbo (title)	79%	74%	91%

Table 8: Result of English textbook evaluation on Baseline, old Q-Gen, Adapter and GPT generated questions

In this context, the GPT-3.5 turbo (title) model got an exact match of 79%, while the GPT-3.5 turbo (context) model attained 63%. The Adapter model marked an exact match of 62%, with the mT-5 baseline model at 53%. The top score was from the Synapse baseline model with an 88% exact match. This high score was expected because the Synapse baseline has been tested and refined with this testbook many times. So, its high score does not fully show how good the question generation method is on its own. When we exclude the Synapse baseline, the GPT-3.5 turbo (title) model performed the best.

As stated before, for the generative system we perform automatic evaluations to see how close the chatbot’s answers were to the correct answers in the testbook. The BERT-Score for the Synapse baseline was 80%. The mT-5 baseline got 55%, the Adapter Model scored 65%, the GPT-3.5 turbo (context) model had 67%, and the GPT-3.5 turbo (title) model got 74%. These trends were also seen in the OpenAI similarity scores, with the Synapse baseline doing better than the other models.

	French Testbook eval		
	Extractive QA	Generative QA	
	Exact Match	BertScore	OpenAI Score
Baseline (mT5-base)	12%	59%	66%
Baseline (Synapse)	19%	80%	89%
Adapter-Based Model	16%	72%	84%
GPT-3.5 turbo (context)	47%	84%	91%
GPT-3.5 turbo (title)	54%	86%	93%

Table 9: Result of French testbook evaluation on Baseline, old Q-Gen, Adapter and GPT generated questions

Table 9 shows the results from the French testbook. In the extractive system evaluation, the GPT-3.5 turbo (title) model leads with a 54% exact match score. The Synapse baseline, surprisingly, only managed 19%, marking a significant drop from its performance with the English testbook. The Adapter model recorded a score of 16%, and the mT-5 baseline got 12%. Following closely behind the leader, the GPT-3.5 turbo (context)

model secured a 47% match. The overall lower scores indicate that the generated French questions definitely needs improvement.

In terms of the BertScore, the GPT-3.5 turbo (title) model tops the list at 86%, with the GPT-3.5 turbo (context) model slightly behind at 84%. Our Adapter model is trailing behind with a score of 72% which is lower than Synapse baseline's score of 80%. When considering the OpenAI similarity score, the Adapter Model scored 84%, whereas the Synapse baseline achieved a commendable 89%.

8 Discussion

In this chapter, we provide a detailed analysis of our model results, discuss our findings, and a summary of our study.

We begin by discussing the performance of our proposed methods in different evaluations. In the English evaluations, the GPT-3.5 turbo (context) showed the best results in the question-to-context similarity test, indicating higher similarity between the generated questions and context. Meanwhile, the GPT-3.5 turbo (title) and the Adapter-based model yielded similar performance levels.

For the textbook evaluations, however, the GPT-3.5 turbo (title) outperformed the GPT-3.5 turbo (context). Notably, the Adapter model was close behind, scoring only 1% less than the GPT-3.5 turbo (context) in terms of exact matches.

Context: Specific security rules are defined by the Security correspondent in agreement with the Country Director or Region SVP and, if necessary, with the support of the Security Directorate. These rules are defined based on the risks identified in the country and local practice. They cover the following areas: * Reception at the airport * List of recommended hotels * Authorised means of transport * Emergency contact in the country These rules, by their nature, apply to staff on a short-term or long-term assignment, expatriates, their families and to service providers in relation to the aspects that concern them and are described in a document called: Country security sheet. Its purpose is to inform and develop the awareness of employees about the rules to be complied with. It also provides practical information (management at customs etc.) on the country in order to facilitate their stay and help them to avoid any errors or blunders. It is reviewed and updated, if necessary, every year except if there are events likely to change the context and the security rules. Once drafted, this sheet is available on the intranet and automatically sent to employees planning to travel to the country.

Title: Country security rules: the country security sheet

Table 10: Sample of context and Title used for generating questions

Initially, we expected that the model with the highest question-to-context similarity would also excel in the textbook evaluation. This is based on the fact that the same

questions and their associated contexts used in the similarity tests were added to the chatbot’s knowledge base for textbook evaluations.

To investigate this discrepancy in performance, we analyzed the questions generated by these models for the same context. Table 10 displays a sample context and title used for question generation. Here, the context was the input for the Adapter and GPT (context) systems, while the GPT-3.5 turbi(title) system only used the title.

Table 11 displays questions derived from both contexts and titles. A review of these examples suggests that the majority align closely with the context’s content. However, the difference between similarity scores and textbook scores becomes clearer upon closer inspection. Questions highlighted in red represent those that might achieve high similarity scores but might not fare as well in the textbook evaluation. The rest of the questions are anticipated to perform well in both assessments as per human judgement.

Adapter-Based: How are the rules of security determined?
What is the purpose of a specific security sheet?
How do they define security rules?

GPT (context): Who defines the specific security rules and who do they consult with?
What areas do these security rules cover?
How often is the Country security sheet reviewed and updated?

GPT (title): What is the purpose of the Country security sheet?
Who is responsible for defining the specific security rules for a country?
How often is the Country security sheet reviewed and updated?

Table 11: Example of generated questions from English context and titles shown in Table 10

Upon examining the questions in Table 11 in relation to the context presented in Table 10, it is evident that each question is closely related to the context. This correlation is further reinforced by our similarity test results. The similarity scores, derived from embeddings and calculated using cosine similarity, affirm this connection between the questions and the context.

However, when we move to the textbook evaluation, which aims to gauge how effec-

tively these questions guide a QA system towards the right answer for a user’s query, the nature of the questions becomes even more critical. Questions highlighted in red from the Adapter and GPT-3.5 turbo(context), with nonspecific terms such as ‘they’, ‘these’, or ‘a specific security’, pose challenges. Their generalized phrasing makes it difficult for the system to accurately match user queries to correct answers. Likewise, the other questions, characterized by clearer terms like ‘Country security sheet’, are more direct. This clarity promotes better alignment in the system, leading to more exact matches during the textbook evaluation.

Thus, although the GPT-3.5 turbo (context) performs well in question-to-context similarity the questions generated from this system does not turn out to be as helpful in the textbook evaluation. Additionally, in terms of our main proposed pipeline, the Adapter method, it reflects similar pattern. While it produces questions closely related to the context, they often lack clarity or contain ambiguity, diminishing its effectiveness in the textbook assessment.

In the French evaluations, the GPT-3.5 (context) model once again showed the best performance in the question-to-context similarity. This is evident from the questions listed in Table 13 and the context provided in Table 12. The questions closely match the given context, which is clear from the significant overlap of tokens, resulting in higher similarity scores.

Context: Le prix de votre carte TaM est de 5 euros. En cas de perte ou de vol de votre carte, des frais de dossier de 3,50 € seront nécessaires à l’établissement d’un nouveau contrat et 5 € pour la création d’une nouvelle carte, soit 8,50 €.

Title: prix de la carte TaM.

Table 12: Sample of context and Title used for generating questions

The GPT-3.5 (title) and Adapter Model scored slightly lower in similarity, but they still performed better than the baseline models. By examining Tables 12 and 13, we see that the questions produced by these models include many of the same tokens from the context. This similarity explains their decent scores.

The performance of the GPT-3.5 (context) model in the French evaluation is consistent

with its results in the English tests. The GPT-3.5 (title) model performed the best, with the Adapter model not far behind.

Adapter-Based: Combien de frais d’ dossier sont nécessaires pour la création du contrat en cas de perte ou vol de votre
De quel montant est le prix de votre carte TaM?
What prix de votre carte TaM if in case

GPT (context): Combien coûte la carte TaM?
Que se passe-t-il en cas de perte ou de vol de la carte TaM?
À quoi correspondent les frais de 3,50 € mentionnés dans le texte ?

GPT (title): Quel est le prix de la carte TaM?
Comment est déterminé le prix de la carte TaM?
Où peut-on acheter la carte TaM au prix indiqué?

Table 13: Example of generated questions from French context and titles shown in Table 12

Looking at Table 13, we can identify some problems with the questions generated by the Adapter model for the French context. While the questions have words from the context, they are often not well-constructed. For example, one question is not clear and is incomplete, and another includes noise. As noted in Section 6.1.5, our model relies on language-specific adapters for each language. The task adapter has only been fine-tuned on the English QG dataset. This might be why it struggles when generating questions in French. For instance, the model generates the English word “What” instead of its French counterpart “Quel”. In another case, the model starts a complex question but does not finish it. These issues suggest that while the similarity scores are decent due to overlapping tokens, these questions might not be effective when used in the textbook evaluation, as they might not correctly link user queries to the right answers.

In summary, after evaluating the quality of questions, we found that the GPT-3.5 turbo (title) system consistently delivered the best results across both evaluations. While the Adapter model did not outperform the GPT models, it still surpassed the other two baseline methods, indicating its effectiveness. It is worth mentioning that the GPT-3.5

turbo model from OpenAI is considerably larger than our Adapter model and has had the advantage of extensive resources and training at OpenAI. Directly comparing our Adapter model to the GPT model might not be a fair assessment. Our primary goal was for the Adapter model to surpass the baselines and, more importantly, to improve upon the existing model at Synapse. In this respect, our model achieved its objectives for the most part. Appendix B contains more examples of the generated questions and contexts for both French and English.

9 Conclusion

From our research, we can conclude that cross-lingual transfer learning using adapters is effective for multilingual question generation, but it has some challenges. The Adapter pipeline we used gives satisfactory results and usually does better than the baseline scores. It is also built in a way that makes it simple to add more languages by just adding new language adapters. This AQG system produces better questions than the system we had before, which shows our project was successful. However, as discussed in Chapter 8, the Adapter model sometimes makes mistakes, especially with languages other than English and in cases where there isn't enough data to train the language adapters properly. Because of this, the LLM Langchain pipeline we made works better in most cases due to its versatility and easy-to-setup nature.

The LLM Langchain pipeline we developed incorporates the GPT-3.5 turbo model. This pipeline was instrumental in enhancing the performance of the DearBot chatbot system at Synapse. Thanks to OpenAI, the extensive training and large number of parameters in this model increase its ability for natural language understanding and generation. With the right prompts, it performs effectively at its task.

Limitations The adapter-based system had its limitations. Due to a lack of computational resources, especially GPUs, we were limited in exploring more languages. This was particularly true for languages with few resources, which would have given us useful information about the system's flexibility. We also did not have enough multilingual QG datasets, so we could not build an adapter pipeline that only used a task adapter trained with a multilingual QG dataset. Additionally, not having specific metrics for question generation made it difficult to judge the quality of questions from our models.

In conclusion, based on our observations and research, we suggest some improvements to our Adapter pipeline. The language adapter training was good for working with different languages. We think that using wiki data along with specific topic data when training the language adapter could make the model better. This approach would help the adapter understand language and also know more about specific topics, improving the chatbot's performance.

10 Bibliography

References

- [1] T. Abhishek, S. Sagare, B. Singh, A. Sharma, M. Gupta, and V. Varma. Xalign: Cross-lingual fact-to-text alignment and generation for low-resource languages. In *Companion Proceedings of the Web Conference 2022, WWW '22*, page 171–175, New York, NY, USA, 2022. Association for Computing Machinery. ISBN 9781450391306. doi: 10.1145/3487553.3524265. URL <https://doi.org/10.1145/3487553.3524265>.
- [2] M. Agarwal and P. Mannem. Automatic gap-fill question generation from text books. In *Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 56–64, Portland, Oregon, June 2011. Association for Computational Linguistics. URL <https://aclanthology.org/W11-1407>.
- [3] M. Agarwal, R. Shah, and P. Mannem. Automatic question generation using discourse cues. In *Proceedings of the Sixth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 1–9, Portland, Oregon, June 2011. Association for Computational Linguistics. URL <https://aclanthology.org/W11-1401>.
- [4] H. Ali, Y. Chali, and S. A. Hasan. Automatic question generation from sentences. In *Actes de la 17e conférence sur le Traitement Automatique des Langues Naturelles. Articles courts*, pages 213–218, Montréal, Canada, July 2010. ATALA. URL <https://aclanthology.org/2010.jeptalnrecital-court.36>.
- [5] R. B. Allen. Several studies on natural language and back-propagation. In *Proceedings of the IEEE First International Conference on Neural Networks*, volume 2, page 341. Citeseer, 1987.
- [6] J. Ba, J. Kiros, and G. Hinton. Layer normalization. 07 2016.
- [7] A. Bagga and B. Baldwin. Entity-based cross-document coreferencing using the vector space model. In *36th Annual Meeting of the Association for Computa-*

- tional Linguistics and 17th International Conference on Computational Linguistics, Volume 1*, pages 79–85, Montreal, Quebec, Canada, Aug. 1998. Association for Computational Linguistics. doi: 10.3115/980845.980859. URL <https://aclanthology.org/P98-1012>.
- [8] L. Becker, S. Basu, and L. Vanderwende. Mind the gap: Learning to choose gaps for question generation. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 742–751, Montréal, Canada, June 2012. Association for Computational Linguistics. URL <https://aclanthology.org/N12-1092>.
- [9] A. L. Berger, S. A. Della Pietra, and V. J. Della Pietra. A maximum entropy approach to natural language processing. *Computational Linguistics*, 22(1):39–71, 1996. URL <https://aclanthology.org/J96-1002>.
- [10] M. Blšták and V. Rozinajova. Automatic question generation based on sentence structure analysis using machine learning approach. *Natural Language Engineering*, 28:1–31, 06 2021. doi: 10.1017/S1351324921000139.
- [11] T. Brants. TnT – a statistical part-of-speech tagger. In *Sixth Applied Natural Language Processing Conference*, pages 224–231, Seattle, Washington, USA, Apr. 2000. Association for Computational Linguistics. doi: 10.3115/974147.974178. URL <https://aclanthology.org/A00-1031>.
- [12] J. Brown, G. Frishkoff, and M. Eskenazi. Automatic question generation for vocabulary assessment. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*, pages 819–826, Vancouver, British Columbia, Canada, Oct. 2005. Association for Computational Linguistics. URL <https://aclanthology.org/H05-1103>.
- [13] T. Brown, B. Mann, N. Ryder, M. Subbiah, J. D. Kaplan, P. Dhariwal, A. Nee-lakantan, P. Shyam, G. Sastry, A. Askell, S. Agarwal, A. Herbert-Voss, G. Krueger, T. Henighan, R. Child, A. Ramesh, D. Ziegler, J. Wu, C. Winter, C. Hesse, M. Chen,

- E. Sigler, M. Litwin, S. Gray, B. Chess, J. Clark, C. Berner, S. McCandlish, A. Radford, I. Sutskever, and D. Amodei. Language models are few-shot learners. In H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 1877–1901. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper_files/paper/2020/file/1457c0d6bfcb4967418bfb8ac142f64a-Paper.pdf.
- [14] Z. Chai and X. Wan. Learning to ask more: Semi-autoregressive sequential question generation under dual-graph interaction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 225–237, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.21. URL <https://aclanthology.org/2020.acl-main.21>.
- [15] Y. Chali and S. Golestanirad. Ranking automatically generated questions using common human queries. In *Proceedings of the 9th International Natural Language Generation conference*, pages 217–221, Edinburgh, UK, Sept. 5-8 2016. Association for Computational Linguistics. doi: 10.18653/v1/W16-6635. URL <https://aclanthology.org/W16-6635>.
- [16] Y. Chali and S. A. Hasan. Towards topic-to-question generation. *Computational Linguistics*, 41(1):1–20, Mar. 2015. doi: 10.1162/COLI_a_00206. URL <https://aclanthology.org/J15-1001>.
- [17] W. Chen. Aist, g., mostow, j.: Generating questions automatically from informational text. In *Proceedings of the 2nd Workshop on Question Generation (AIED 2009)*, pages 17–24, 2009.
- [18] K. Cho, B. van Merriënboer, C. Gulcehre, D. Bahdanau, F. Bougares, H. Schwenk, and Y. Bengio. Learning phrase representations using RNN encoder–decoder for statistical machine translation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1724–1734, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1179. URL <https://aclanthology.org/D14-1179>.

- [19] E. Choi, H. He, M. Iyyer, M. Yatskar, W.-t. Yih, Y. Choi, P. Liang, and L. Zettlemoyer. QuAC: Question answering in context. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2174–2184, Brussels, Belgium, Oct.-Nov. 2018. Association for Computational Linguistics. doi: 10.18653/v1/D18-1241. URL <https://aclanthology.org/D18-1241>.
- [20] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov. Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8440–8451, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.747. URL <https://aclanthology.org/2020.acl-main.747>.
- [21] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL <https://aclanthology.org/N19-1423>.
- [22] X. Du and C. Cardie. Identifying where to focus in reading comprehension for neural question generation. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 2067–2073, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1219. URL <https://aclanthology.org/D17-1219>.
- [23] X. Du and C. Cardie. Harvesting paragraph-level question-answer pairs from Wikipedia. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1907–1917, Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-1177. URL <https://aclanthology.org/P18-1177>.

- [24] X. Du, J. Shao, and C. Cardie. Learning to ask: Neural question generation for reading comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1342–1352, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1123. URL <https://aclanthology.org/P17-1123>.
- [25] N. Duan, D. Tang, P. Chen, and M. Zhou. Question generation for question answering. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 866–874, Copenhagen, Denmark, Sept. 2017. Association for Computational Linguistics. doi: 10.18653/v1/D17-1090. URL <https://aclanthology.org/D17-1090>.
- [26] A. Fan, Y. Jernite, E. Perez, D. Grangier, J. Weston, and M. Auli. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy, July 2019. Association for Computational Linguistics. doi: 10.18653/v1/P19-1346. URL <https://aclanthology.org/P19-1346>.
- [27] M. Flor and B. Riordan. A semantic role-based approach to open-domain automatic question generation. In *Proceedings of the Thirteenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 254–263, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/W18-0530. URL <https://aclanthology.org/W18-0530>.
- [28] M. Fuadi and A. D. Wibawa. Automatic question generation from indonesian texts using text-to-text transformers. In *2022 International Conference on Electrical and Information Technology (IEIT)*, pages 84–89, 2022. doi: 10.1109/IEIT56384.2022.9967858.
- [29] F. A. Gers, Jürgen Schmidhuber, and F. Cummins. Learning to forget: Continual prediction with lstm learning to forget: Continual prediction with lstm. 1999. URL <https://api.semanticscholar.org/CorpusID:7498449>.

- [30] B. Ghanem, L. Lutz Coleman, J. Rivard Dexter, S. von der Ohe, and A. Fyshe. Question generation for reading comprehension assessment by modeling how and what to ask. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 2131–2146, Dublin, Ireland, May 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-acl.168. URL <https://aclanthology.org/2022.findings-acl.168>.
- [31] D. Goldhahn, T. Eckart, and U. Quasthoff. Building large monolingual dictionaries at the Leipzig corpora collection: From 100 to 200 languages. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC’12)*, pages 759–765, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA). URL http://www.lrec-conf.org/proceedings/lrec2012/pdf/327_Paper.pdf.
- [32] A. C. Graesser and N. K. Person. Question asking during tutoring. *American Educational Research Journal*, 31(1):104–137, 1994. doi: 10.3102/00028312031001104. URL <https://doi.org/10.3102/00028312031001104>.
- [33] W. Han, B. Pang, and Y. N. Wu. Robust transfer learning with pretrained language models through adapters. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, pages 854–861, Online, Aug. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-short.108. URL <https://aclanthology.org/2021.acl-short.108>.
- [34] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778, 2016. doi: 10.1109/CVPR.2016.90.
- [35] R. He, L. Liu, H. Ye, Q. Tan, B. Ding, L. Cheng, J. Low, L. Bing, and L. Si. On the effectiveness of adapter-based tuning for pretrained language model adaptation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language*

- Processing (Volume 1: Long Papers)*, pages 2208–2222, Online, Aug. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.172. URL <https://aclanthology.org/2021.acl-long.172>.
- [36] M. Heilman and N. A. Smith. Question generation via overgenerating transformations and ranking. 2009. URL <https://api.semanticscholar.org/CorpusID:14631309>.
- [37] M. Heilman and N. A. Smith. Extracting simplified statements for factual question generation. 2010. URL <https://api.semanticscholar.org/CorpusID:2123424>.
- [38] M. Heilman and N. A. Smith. Good question! statistical ranking for question generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 609–617, Los Angeles, California, June 2010. Association for Computational Linguistics. URL <https://aclanthology.org/N10-1086>.
- [39] J. Henderson, S. Ruder, et al. Compacter: Efficient low-rank hypercomplex adapter layers. In *Advances in Neural Information Processing Systems*, 2021.
- [40] N. Houlsby, A. Giurgiu, S. Jastrzebski, B. Morrone, Q. De Laroussilhe, A. Gesmundo, M. Attariyan, and S. Gelly. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR, 2019.
- [41] Y. Huang and L. He. Automatic generation of short answer questions for reading comprehension assessment. *Natural Language Engineering*, 22(3):457–489, 2016.
- [42] K. S. Jones. A statistical interpretation of term specificity and its application in retrieval. *J. Documentation*, 60:493–502, 2021. URL <https://api.semanticscholar.org/CorpusID:2996187>.
- [43] A. I. Kadhim, Y.-N. Cheah, I. A. Hieder, and R. A. Ali. Improving tf-idf with singular value decomposition (svd) for feature extraction on twitter. 2017. URL <https://api.semanticscholar.org/CorpusID:126375426>.

- [44] S. Kalady, A. Elikkottil, and R. Das. Natural language question generation using syntax and keywords. 2010.
- [45] R. Karimi Mahabadi, S. Ruder, M. Dehghani, and J. Henderson. Parameter-efficient multi-task fine-tuning for transformers via shared hypernetworks. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 565–576, Online, Aug. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.acl-long.47. URL <https://aclanthology.org/2021.acl-long.47>.
- [46] Y. Kim, H. Lee, J. Shin, and K. Jung. Improving neural question generation using answer separation. In *Proceedings of the Thirty-Third AAAI Conference on Artificial Intelligence and Thirty-First Innovative Applications of Artificial Intelligence Conference and Ninth AAAI Symposium on Educational Advances in Artificial Intelligence, AAAI’19/IAAI’19/EAAI’19*. AAAI Press, 2019. ISBN 978-1-57735-809-1. doi: 10.1609/aaai.v33i01.33016602. URL <https://doi.org/10.1609/aaai.v33i01.33016602>.
- [47] T. Kočiský, J. Schwarz, P. Blunsom, C. Dyer, K. M. Hermann, G. Melis, and E. Grefenstette. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328, 2018. doi: 10.1162/tacl_a.00023. URL <https://aclanthology.org/Q18-1023>.
- [48] G. Kumar, R. Banchs, and L. F. D’Haro. RevUP: Automatic gap-fill question generation from educational texts. In *Proceedings of the Tenth Workshop on Innovative Use of NLP for Building Educational Applications*, pages 154–161, Denver, Colorado, June 2015. Association for Computational Linguistics. doi: 10.3115/v1/W15-0618. URL <https://aclanthology.org/W15-0618>.
- [49] I. Labutov, S. Basu, and L. Vanderwende. Deep questions without deep understanding. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Lan-*

- guage Processing (Volume 1: Long Papers)*, pages 889–898, Beijing, China, July 2015. Association for Computational Linguistics. doi: 10.3115/v1/P15-1086. URL <https://aclanthology.org/P15-1086>.
- [50] C.-H. Lee, A. Siddhant, V. Ratnakar, and M. Johnson. DOCmT5: Document-level pretraining of multilingual language models. In *Findings of the Association for Computational Linguistics: NAACL 2022*, pages 425–437, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.findings-naacl.32. URL <https://aclanthology.org/2022.findings-naacl.32>.
- [51] W. Lehnert. *A Conceptual Theory of Question Answering*, page 651–657. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1986. ISBN 0934613117.
- [52] D. Lindberg, F. Popowich, J. Nesbit, and P. Winne. Generating natural language questions to support learning on-line. In *Proceedings of the 14th European Workshop on Natural Language Generation*, pages 105–114, Sofia, Bulgaria, Aug. 2013. Association for Computational Linguistics. URL <https://aclanthology.org/W13-2114>.
- [53] B. Liu, M. Zhao, D. Niu, K. Lai, Y. He, H. Wei, and Y. Xu. Learning to generate questions by learning what not to generate. In *The World Wide Web Conference, WWW ’19*, page 1106–1118, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450366748. doi: 10.1145/3308558.3313737. URL <https://doi.org/10.1145/3308558.3313737>.
- [54] B. Liu, H. Wei, D. Niu, H. Chen, and Y. He. Asking questions the human way: Scalable question-answer generation from text corpus. In *Proceedings of The Web Conference 2020, WWW ’20*, page 2032–2043, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450370233. doi: 10.1145/3366423.3380270. URL <https://doi.org/10.1145/3366423.3380270>.
- [55] M. Liu, R. A. Calvo, and V. Rus. Automatic question generation for literature review writing support. In *International Conference on Intelligent Tutoring Systems*, 2010. URL <https://api.semanticscholar.org/CorpusID:13917826>.

- [56] Y. Liu, J. Gu, N. Goyal, X. Li, S. Edunov, M. Ghazvininejad, M. Lewis, and L. Zettlemoyer. Multilingual denoising pre-training for neural machine translation. *Transactions of the Association for Computational Linguistics*, 8:726–742, 2020. doi: 10.1162/tacl_a_00343. URL <https://aclanthology.org/2020.tacl-1.47>.
- [57] T. Luong, H. Pham, and C. D. Manning. Effective approaches to attention-based neural machine translation. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1412–1421, Lisbon, Portugal, Sept. 2015. Association for Computational Linguistics. doi: 10.18653/v1/D15-1166. URL <https://aclanthology.org/D15-1166>.
- [58] K. Mazidi and R. D. Nielsen. Linguistic considerations in automatic question generation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 321–326, Baltimore, Maryland, June 2014. Association for Computational Linguistics. doi: 10.3115/v1/P14-2053. URL <https://aclanthology.org/P14-2053>.
- [59] T. Mikolov, M. Karafiát, L. Burget, J. Cernocký, and S. Khudanpur. Recurrent neural network based language model. In *Interspeech*, volume 2, pages 1045–1048. Makuhari, 2010.
- [60] R. Mitkov and L. A. Ha. Computer-aided generation of multiple-choice tests. In *Proceedings of the HLT-NAACL 03 Workshop on Building Educational Applications Using Natural Language Processing*, pages 17–22, 2003. URL <https://aclanthology.org/W03-0203>.
- [61] N. Mostafazadeh, I. Misra, J. Devlin, M. Mitchell, X. He, and L. Vanderwende. Generating natural questions about an image. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1802–1813, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1170. URL <https://aclanthology.org/P16-1170>.
- [62] J. Mostow and W. Chen. Generating instruction automatically for the reading

- strategy of self-questioning. In *International Conference on Artificial Intelligence in Education*, 2009. URL <https://api.semanticscholar.org/CorpusID:16005144>.
- [63] X. Mou, C. Yang, M. Yu, B. Yao, X. Guo, S. Potdar, and H. Su. Narrative question answering with cutting-edge open-domain QA techniques: A comprehensive study. *Transactions of the Association for Computational Linguistics*, 9:1032–1046, 2021. doi: 10.1162/tacl_a_00411. URL <https://aclanthology.org/2021.tacl-1.61>.
- [64] N. Mulla and P. Gharpure. Automatic question generation: a review of methodologies, datasets, evaluation metrics, and applications. *Progress in Artificial Intelligence*, 12, 01 2023. doi: 10.1007/s13748-023-00295-9.
- [65] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, and L. Deng. Ms marco: A human-generated machine reading comprehension dataset. 2016.
- [66] Y. Pan, M. Ma, B. Pflugfelder, and G. Groh. How to build robust faq chatbot with controllable question generator? *ArXiv*, abs/2112.03007, 2021. URL <https://api.semanticscholar.org/CorpusID:244909063>.
- [67] J. Pennington, R. Socher, and C. Manning. GloVe: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, Oct. 2014. Association for Computational Linguistics. doi: 10.3115/v1/D14-1162. URL <https://aclanthology.org/D14-1162>.
- [68] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana, June 2018. Association for Computational Linguistics. doi: 10.18653/v1/N18-1202. URL <https://aclanthology.org/N18-1202>.
- [69] J. Pfeiffer, I. Vulić, I. Gurevych, and S. Ruder. MAD-X: An Adapter-Based Framework for Multi-Task Cross-Lingual Transfer. In *Proceedings of the 2020 Conference*

- on *Empirical Methods in Natural Language Processing (EMNLP)*, pages 7654–7673, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.617. URL <https://aclanthology.org/2020.emnlp-main.617>.
- [70] J. Pfeiffer, A. Kamath, A. Rücklé, K. Cho, and I. Gurevych. AdapterFusion: Non-destructive task composition for transfer learning. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pages 487–503, Online, Apr. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.eacl-main.39. URL <https://aclanthology.org/2021.eacl-main.39>.
- [71] P. Piwek and S. Stoyanchev. Question generation in the coda project. 2010. URL <https://api.semanticscholar.org/CorpusID:30358008>.
- [72] T. Poibeau and L. Kosseim. Proper name extraction from non-journalistic texts. volume 37, pages 144–157, 01 2000.
- [73] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.
- [74] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu. Exploring the limits of transfer learning with a unified text-to-text transformer. *The Journal of Machine Learning Research*, 21(1):5485–5551, 2020.
- [75] P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas, Nov. 2016. Association for Computational Linguistics. doi: 10.18653/v1/D16-1264. URL <https://aclanthology.org/D16-1264>.
- [76] P. Rajpurkar, R. Jia, and P. Liang. Know what you don’t know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789,

- Melbourne, Australia, July 2018. Association for Computational Linguistics. doi: 10.18653/v1/P18-2124. URL <https://aclanthology.org/P18-2124>.
- [77] S. E. Robertson. Understanding inverse document frequency: on theoretical arguments for idf. *J. Documentation*, 60:503–520, 2004. URL <https://api.semanticscholar.org/CorpusID:8864928>.
- [78] O. Rokhlenko and I. Szpektor. Generating synthetic comparable questions for news articles. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 742–751, Sofia, Bulgaria, Aug. 2013. Association for Computational Linguistics. URL <https://aclanthology.org/P13-1073>.
- [79] A. Rücklé, G. Geigle, M. Glockner, T. Beck, J. Pfeiffer, N. Reimers, and I. Gurevych. AdapterDrop: On the efficiency of adapters in transformers. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 7930–7946, Online and Punta Cana, Dominican Republic, Nov. 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.emnlp-main.626. URL <https://aclanthology.org/2021.emnlp-main.626>.
- [80] V. Rus, B. Wyse, P. Piwek, M. Lintean, S. Stoyanchev, and C. Moldovan. The first question generation shared task evaluation challenge. In *Proceedings of the 6th International Natural Language Generation Conference*. Association for Computational Linguistics, July 2010. URL <https://aclanthology.org/W10-4234>.
- [81] I. A. Sag and D. Flickinger. Generating questions with deep reversible grammars.
- [82] A. See, P. J. Liu, and C. D. Manning. Get to the point: Summarization with pointer-generator networks. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1073–1083, Vancouver, Canada, July 2017. Association for Computational Linguistics. doi: 10.18653/v1/P17-1099. URL <https://aclanthology.org/P17-1099>.
- [83] I. V. Serban, A. García-Durán, C. Gulcehre, S. Ahn, S. Chandar, A. Courville, and

- Y. Bengio. Generating factoid questions with recurrent neural networks: The 30M factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 588–598, Berlin, Germany, Aug. 2016. Association for Computational Linguistics. doi: 10.18653/v1/P16-1056. URL <https://aclanthology.org/P16-1056>.
- [84] S. Shakeri, N. Constant, M. Kale, and L. Xue. Towards zero-shot multilingual synthetic question and answer generation for cross-lingual reading comprehension. In *Proceedings of the 14th International Conference on Natural Language Generation*, pages 35–45, Aberdeen, Scotland, UK, Aug. 2021. Association for Computational Linguistics. URL <https://aclanthology.org/2021.inlg-1.4>.
- [85] L. Song, Z. Wang, and W. Hamza. A unified query-based generative model for question generation and question answering. *arXiv preprint arXiv:1709.01058*, 2017.
- [86] A. C. Stickland and I. Murray. Bert and pals: Projected attention layers for efficient adaptation in multi-task learning. In *International Conference on Machine Learning*, 2019. URL <https://api.semanticscholar.org/CorpusID:59608712>.
- [87] I. Sutskever, O. Vinyals, and Q. V. Le. Sequence to sequence learning with neural networks. In *Proceedings of the 27th International Conference on Neural Information Processing Systems - Volume 2*, NIPS’14, page 3104–3112, Cambridge, MA, USA, 2014. MIT Press.
- [88] D. Tang, N. Duan, T. Qin, Z. Yan, and M. Zhou. Question answering and question generation as dual tasks. *arXiv preprint arXiv:1706.02027*, 2017.
- [89] A. VARGA and L. Ha. Wlv: A question generation system for the qgstec 2010 task b. *QG2010: The third workshop on Question Generation*, 06 2010.
- [90] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, L. u. Kaiser, and I. Polosukhin. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates,

- Inc., 2017. URL https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf.
- [91] S. Wang, Z. Wei, Z. Fan, Z. Huang, W. Sun, Q. Zhang, and X. Huang. PathQG: Neural question generation from facts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 9066–9075, Online, Nov. 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.729. URL <https://aclanthology.org/2020.emnlp-main.729>.
- [92] W. Wang, T. Hao, and W. Liu. Automatic question generation for learning evaluation in medicine. In *Advances in Web Based Learning–ICWL 2007: 6th International Conference Edinburgh, UK, August 15-17, 2007 Revised Papers 6*, pages 242–251. Springer, 2008.
- [93] J. Weizenbaum. Eliza—a computer program for the study of natural language communication between man and machine. *Commun. ACM*, 9(1):36–45, jan 1966. ISSN 0001-0782. doi: 10.1145/365153.365168. URL <https://doi.org/10.1145/365153.365168>.
- [94] N. Wiwatbutsiri, A. Suchato, P. Punyabukkana, and N. Tuaycharoen. Question generation in the thai language using mt5. In *2022 19th International Joint Conference on Computer Science and Software Engineering (JCSSE)*, pages 1–6, 2022. doi: 10.1109/JCSSE54890.2022.9836271.
- [95] L. Xue, N. Constant, A. Roberts, M. Kale, R. Al-Rfou, A. Siddhant, A. Barua, and C. Raffel. mT5: A massively multilingual pre-trained text-to-text transformer. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 483–498, Online, June 2021. Association for Computational Linguistics. doi: 10.18653/v1/2021.naacl-main.41. URL <https://aclanthology.org/2021.naacl-main.41>.
- [96] X. Yao. Question generation with minimal recursion semantics. 2010. URL <https://api.semanticscholar.org/CorpusID:18796690>.

- [97] J. Yu, W. Liu, S. Qiu, Q. Su, K. Wang, X. Quan, and J. Yin. Low-resource generation of multi-hop reasoning questions. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6729–6739, Online, July 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.acl-main.601. URL <https://aclanthology.org/2020.acl-main.601>.
- [98] X. Yuan, T. Wang, C. Gulcehre, A. Sordoni, P. Bachman, S. Zhang, S. Subramanian, and A. Trischler. Machine comprehension by text-to-text neural question generation. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 15–25, Vancouver, Canada, Aug. 2017. Association for Computational Linguistics. doi: 10.18653/v1/W17-2603. URL <https://aclanthology.org/W17-2603>.
- [99] T. Zhang, V. Kishore, F. Wu, K. Q. Weinberger, and Y. Artzi. Bertscore: Evaluating text generation with bert, 2020.
- [100] Z. Zheng, X. Si, E. Chang, and X. Zhu. K2Q: Generating natural language questions from keywords with user refinements. In *Proceedings of 5th International Joint Conference on Natural Language Processing*, pages 947–955, Chiang Mai, Thailand, Nov. 2011. Asian Federation of Natural Language Processing. URL <https://aclanthology.org/I11-1106>.
- [101] Q. Zhou, N. Yang, F. Wei, C. Tan, H. Bao, and M. Zhou. Neural question generation from text: A preliminary study. In *Natural Language Processing and Chinese Computing*, 2017. URL <https://api.semanticscholar.org/CorpusID:9745861>.

A Prompts

A.1 Prompt + Context

As a question generator system, your objective is to analyze a given text excerpt and produce distinct and relevant questions that can be answered using the provided information. It is crucial that the questions are directly related to the text and do not include any irrelevant details. The generated text should be in the same language as the given text excerpt. Under no circumstances should you generate questions that are unrelated to the given text. If the text is insufficient for generating questions, simply state ‘NaN’ without providing any additional information. You are explicitly forbidden from modifying or expanding the text in an attempt to create more context. The questions should cover a range of types like ‘what’, ‘how’, ‘why’, ‘where’, etc. Remember, do not add, modify, or infer any context beyond what is explicitly stated in the text.

Extract of text : {context}.

A.2 Prompt + Title

Your goal as a question generator system is to carefully analyze a given text excerpt, which can be in the form of titles, phrases, or questions, and generate distinct and relevant questions based on that text. The generated text should be in the same language as the given text excerpt. If the text provided is a title or phrase, you should generate questions using that text. If the input is already a question, your task is to rephrase it. It is essential that the questions you generate are directly related to the text and do not include any irrelevant details. You must not generate questions that are unrelated to the given text, and you are not allowed to modify or expand the text to create more context. Your sole responsibility is to generate questions, ensuring they cover a range of types such as ‘what’, ‘how’, ‘why’, ‘where’, etc.

Extract of text : {title}.

B Examples of Generated Questions

B.1 Adapter-Based System

B.1.1 English Example

Context: SYSTRA has a duty to communicate all of the necessary information in order to ensure that all of the stakeholders (employees, employees families, service providers, etc.) are familiar with the security rules/instructions. Communication relates to the following subjects and takes place via the channels listed below: ITEM ISSUER CHANNELS Country security sheets Security Directorate Intranet Security recommendation e-mail for the country concerned Thematic sheets Security Directorate Intranet Link in the country sheet Security correspondent Security Guide for business travellers Security Directorate Intranet Security recommendation e-mail for the country concerned Country/site security instructions Security Directorate Intranet (country or project sheet) Security correspondent Security alerts Security Directorate Country Director.

Generated Questions:

1. What do SYSTRA communicate?
2. What does SYSTRA communicate to the stakeholders?
3. How does SYSTRA communicate all of the security requirements?

B.1.2 French Example

Context: Vous pouvez contacter Allo TaM au 04-67-22-87-87, du Lundi au Vendredi de 7h30 à 18h30 et le Samedi matin de 9h à 12h.

Generated Questions:

1. Comment se contacter Allo TaM au 04-67-22-87-87?
2. Quel est le nouveau mot du dernier jour des heures de la seconde minute?
3. Comment contacter Allo TaM??

B.2 LLM Langchain System

B.2.1 Prompt + Context (English)

Context: SYSTRA has a duty to communicate all of the necessary information in order to ensure that all of the stakeholders (employees, employees families, service providers, etc.) are familiar with the security rules/instructions. Communication relates to the following subjects and takes place via the channels listed below: ITEM ISSUER CHANNELS Country security sheets Security Directorate Intranet Security recommendation e-mail for the country concerned Thematic sheets Security Directorate Intranet Link in the country sheet Security correspondent Security Guide for business travellers Security Directorate Intranet Security recommendation e-mail for the country concerned Country/site security instructions Security Directorate Intranet (country or project sheet) Security correspondent Security alerts Security Directorate Country Director

Generated Questions:

1. What is the role of SYSTRA in communicating security rules/instructions?
2. What are the different channels through which SYSTRA communicates security-related information?
3. Who issues the country security sheets and security alerts at SYSTRA?

B.2.2 Prompt + Context (French)

Context: Vous pouvez contacter Allo TaM au 04-67-22-87-87, du Lundi au Vendredi de 7h30 à 18h30 et le Samedi matin de 9h à 12h.

Generated Questions:

1. À quel numéro peut-on contacter Allo TaM?
2. Quels sont les horaires pour contacter Allo TaM le Vendredi?
3. Est-il possible de joindre Allo TaM le Samedi après-midi?

B.2.3 Prompt + Title (English)

Title: Communication¹³

Generated Questions:

1. What are the subjects that SYSTRA's communication relates to?
2. How does SYSTRA communicate the security rules and instructions to its stakeholders?
3. Who issues the security alerts at SYSTRA and through what channel are they communicated?

B.2.4 Prompt + Title (French)

Title: J'ai perdu la carte TaM, quelles sont les démarches ?¹⁴

Generated Questions:

1. Quelles étapes dois-je suivre après avoir perdu ma carte TaM?
2. Comment puis-je remplacer ma carte TaM perdue?
3. Où dois-je m'adresser pour signaler la perte de ma carte TaM?

¹³Here the title is from the same context that is used in the other two pipelines

¹⁴The title here is in the form of question since some of the client data are FAQ documents for which the questions are labeled as titles