Institute of Computer Science
University of Augsburg

Master Thesis

Multilingual Question Answering in a German Migration Context Using Neural Methods

Steffen Kleinle

Reviewer:	Prof. Dr. Annemarie Friedrich
Second Reviewer:	Prof. Dr. Alexander Knapp
Supervisor:	Dr. Jakob Prange
Matriculation:	1474554
Date:	July 12, 2024

written at the
Chair of Natural Language Understanding with Applications to DH
Prof. Dr. Annemarie Friedrich
Institute of Computer Science
University of Augsburg
86159 Augsburg, Germany

Abstract

Information is imperative for participating in society and everyday life, even more so if someone is immigrating to a new country, possibly forced by war, persecution, or hunger. The lack of information among newcomers, which often revolves around topics such as work and employment, housing, education, language, and other issues, must be resolved to allow for a successful integration. However, information is scattered among the web and print materials, immigration counselors and support offers are usually overbooked and overworked, and German laws and authorities are notoriously bureaucratic and complicated. On top of that, accessing information and communication is made increasingly difficult by the language barrier. Online services and recent advances in natural language processing (NLP) can help to mitigate these issues and bridge the gap between plain textual information and professional human counselors. To this end, we investigate the suitability of question answering (QA) systems in a migration context and put a focus on trustworthiness.

In order to allow for proper training and evaluation of QA systems, we present OMoS-QA, a dataset specifically tailored to this scenario. Questions are automatically generated with an open-weight large language model (LLM) in German and English, and answer sentences are crowdsourced with high agreement from relevant trustworthy documents. We include unanswerable questions, where the answer cannot be found in the paired document, to allow for practical use in real-life application scenarios. To foster the human annotation process, we develop a custom web-based annotation tool that is made available as open-source software. Thus, we show that both NLP techniques leveraging LLMs and crowdsourcing with untrained volunteers can play an important role in facilitating the construction of a dataset when the tasks are modular and restrained. Our dataset consists of 906 high-quality QA pairs in German and English.

With our data, we evaluate the QA capabilities of different approaches in a multilingual immigration context. We only consider extractive QA, as generative approaches are known to suffer from hallucinations and other unfavorable behavior. We focus on open-weight LLMs, namely Mistral-7B, Mixtral-8x7B, and the Llama-3 model family. For comparison, we add results from closed-source GPT-3.5-Turbo and from finetuning experiments with DeBERTa. We evaluate the models in various settings on the German and English OMoS-QA. Most LLMs exhibit high precision and medium recall, which is in line with our focus on trustworthiness. Even providing documents in a different language than the questions does not necessarily hurt and sometimes even improves this performance. Few-shot prompting increases model performance and instruction following of LLMs. Additionally, we investigate the multilingual QA capabilities of neural models in comparison to leveraging machine translation (MT) in Arabic, French, and Ukrainian and find a slight advantage of the latter. Compared to the QA with the original dataset in German and English, a slight loss of performance is still measurable. Finetuned DeBERTa exhibits competitive results, albeit showing more balanced precision and recall. Several models outperform the human agreement, which is inferred from the inter-annotator agreement of our human annotations.

Apart from using our models to elicit answer sentences to questions, we experiment with classifying whether a question is answerable or not given a document. We compare two settings: One where we infer these numbers from sentence-level results and one where we explicitly finetune or prompt models for this task. Explicit prompting with Llama-3-70B exhibits the best results and proves to be capable in detecting unanswerable questions.

Our dataset, the annotation tool, and all our experiments and results are published on GitHub.[1]

[1] `https://github.com/digitalfabrik/integreat-qa-dataset`

Contents

1. Introduction

With the end of the year 2023, worldwide displacement has reached a new record high of more than 117 million forcibly displaced people (United Nations, 2024). Apart from obvious basic needs of food, shelter, and healthcare, information access is a serious problem for many refugees. Information poverty prevents a successful integration and complicates most aspects of life. Recent advances in natural language processing (NLP), especially in regard to large language models (LLMs) and machine translation (MT), present the opportunity to diminish the difficulties of information access. In this work, we explore different NLP techniques to assist with this problem.

1.1 Motivation

The Russian attack on Ukraine in February 2022 and the following and still ongoing war has interrupted the mostly peaceful order in Europe after World War II. This war has already cost the lives of at least 10,000 and injured more than 19,000 civilians, with casualties among military personnel in the hundreds of thousands on both sides.[1] As with most armed conflicts, the lives of many more people are affected: Because of the war, Ukraine has seen more than 5 million internally displaced people (IOM, 2023) and 6.5 million refugees fleeing to other countries, out of which close to 1.2 million refugees have since arrived in Germany.[2] However, this is only the latest wave of immigration that Germany has seen. Since 2015, more than one million war refugees primarily from Syria, Iraq, and Afghanistan have found a temporary or permanent residence in Germany.[3] Apart from war refugees, Germany has also experienced several immigration movements due to economic reasons, for example guest workers from Italy and Turkey in the 1950s and 60s, ethnic Germans from countries in Eastern Europe (so-called Aussiedler) in the 1980s and 90s, and from other European Union countries with free movement more recently.

Integrating these newcomers poses enormous challenges for the host country, its inhabitants, and foremost, the newcomers themselves. While migrants obviously benefit from good integration in the society by finding work, learning the language, and growing their social environment, host countries profit strongly from successful integration as well. Even though migration leads to short-term costs and challenges, e.g., for the housing or labor markets, newcomers enrich and benefit their destination countries in various dimensions. They open up new perspectives and ideas, can increase economic value, and provide opportunities for companies, especially in aging countries with severe labor shortage such as Germany (Koczan et al., 2021). However, successful integration is crucial to enable positive effects for both the host country and the newcomers, even more so, if the relocation is involuntary due to war or persecution.

To this end, access to information about the immigration and the accompanied procedures as well as for the following every-day life is imperative. A failure to satisfy those information needs can have severe effects on the help seeker, up to unemployment, homelessness, or deportation. Since procedures in Germany are often slow and bureaucratic, albeit trying to support those in need, newcomers frequently require help to successfully navigate them. Germany has an established and working support system of official counselors, NGOs, and volunteers providing extensive counseling

[1] https://www.ohchr.org/en/documents/country-reports/two-year-update-protection-civilians-impact-hostilities-civilians-24

[2] https://data.unhcr.org/en/situations/ukraine

[3] https://www.bpb.de/shop/zeitschriften/apuz/312832/vor-dem-5-september

and help to those in need. Additionally, lots of initiatives from authorities and non-profits provide information resources online and in print. However, information is scattered among online and print resources and often outdated, incomplete, or unstructured. It is also often location-specific and focuses on particular aspects. Furthermore, the language barrier substantially complicates the process of acquiring information. As a consequence, it is difficult for newcomers to find helpful and relevant information. At the same time, counselors and volunteers are often overworked with limited time and capabilities to attend to all individuals, even more so in times of big migration movements due to hunger, persecution, or war.

1.2 The Integreat-App and OMoS

Information poverty can have various reasons: Illiteracy, cognitive or visual impairments, or simply unfamiliarity with modern digital technologies. For refugees and newcomers in general, an additional serious obstacle is posed by the language barrier of not speaking the host country's native language or English. Countless organizations, initiatives, and volunteers in Germany have detected the lack of information of individuals as a serious problem and dedicate their work to mitigate this information poverty.[4] [5] [6] Different projects focus on different levels and aspects of the problem and its symptoms, making use of a variety of approaches.

In the following, we describe the Integreat-App, a particularly successful initiative that aims to reduce the information poverty among refugees and newcomers in general by providing multilingual and local information. We then describe OMoS, a holistic idea to extend and advance the Integreat-App by combining digital information and NLP techniques with human expert counseling.

The Integreat-App. The *Integreat-App* is a multilingual information platform developed by *Tür an Tür Digitalfabrik*, a non-profit organization in the migration and integration context based in Augsburg, Germany. It aims to collect and provide trustworthy local information digitally to tackle information poverty among newcomers, asylum seekers, and non-native German speakers to foster the arrival and integration process. The Integreat-App tries to remove the language barrier by supporting more than 30 languages that are highly relevant in the migration context, such as Arabic, Farsi, or Ukrainian. While the Integreat-App started in 2015 as a response in the wake of the so-called *refugee crisis* to help refugees, its focus has since shifted to migration and information access in general. The app covers topics ranging from the migration to and arriving in Germany itself to everyday issues such as language, education, health, or work, among others. The Integreat-App shifts individual information access from analogous brochures and outdated printed manuals to flexible, centralized, and easily accessible online resources.

Since regulation and support offers in Germany are highly decentralized on both state and district level, the Integreat-App follows an approach to directly involve the local administrative districts instead of attempting a one-fits-all solution. In other words, content in the Integreat-App is managed separately by the various districts, at the moment more than 100 and therefore more than a quarter of all German administrative districts and cities. Each district can decide individually on the topics to include, languages to use, and demographic groups to target based on the local situation. Tür an Tür Digitalfabrik provides a generic template with general information on various topics, which most districts use to get started. As a result, the information provided has substantial overlap between the different districts.

[4] `https://deutschland.welcome-app-germany.de`
[5] `https://tuerantuer.de/integrationsprojekte/sprachangebote/deutsch-cafe`
[6] `https://alfa-telefon.de`

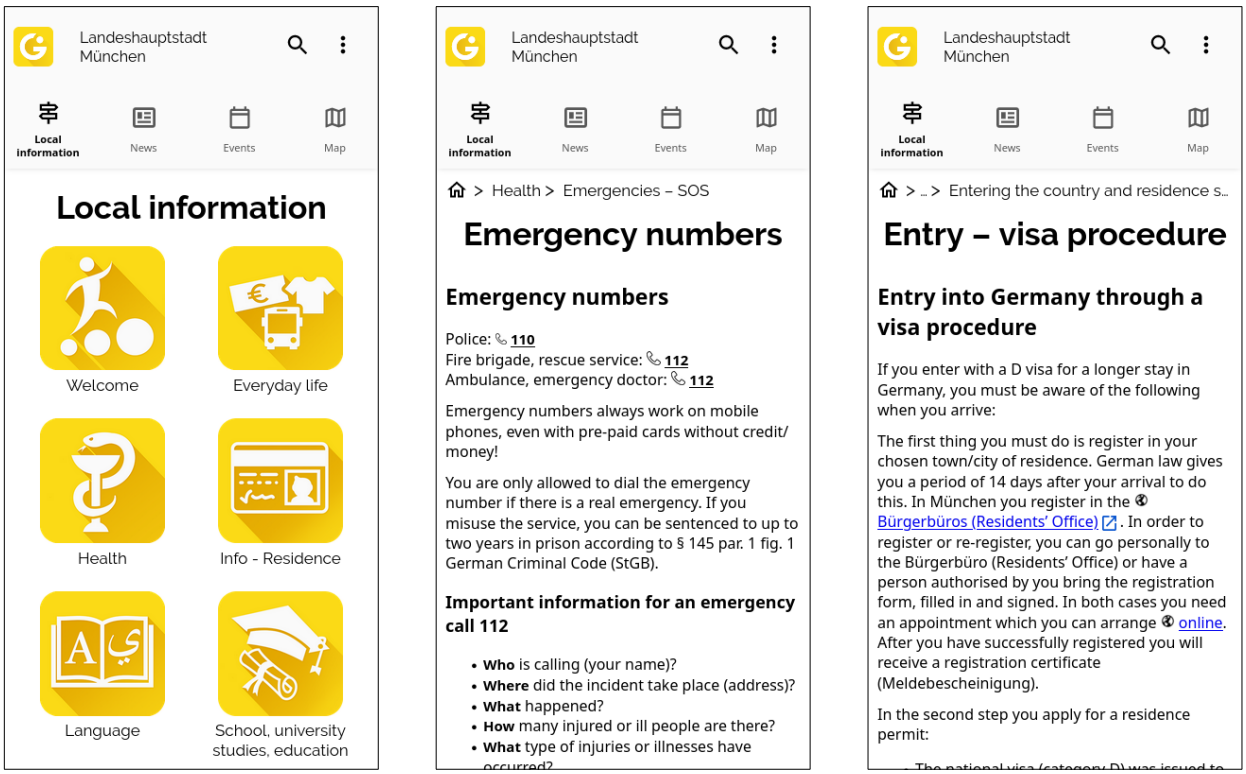

Figure 1.1: Screenshots of the Integreat-App for the city of Munich.

The platform consists of a content management system (CMS) written in Python using the Django framework and a React web and React Native apps for iOS and Android developed in TypeScript. The offer is completed by various support offers for cities and municipalities, such as dedicated translation services, advertisement campaigns, and best-practice guides. The Integreat-App is completely open-source with all source code published on GitHub.[7] All content is licensed under the Creative Commons CC BY 4.0 license.[8] Screenshots of the Integreat-App and examples of available information can be seen in Fig. 1.1.

OMoS. While the Integreat-App is an important step against information poverty among newcomers, it can only be part of a bigger solution. Even though it allows for fast and low-threshold information access with partly removing the language barrier, it is not feasible to represent all information for every possible individual situation in a (textual) database and in every possible language. Complex circumstances and regulation, illiteracy, disabilities, or missing knowledge about structures and rules in the host country can further complicate accessing and understanding the right information. Hence, manual counselling by human experts is still necessary to provide in-depth help on more difficult cases. However, manual counselling is resource-intensive, inflexible, and suffers from additional difficulties such as the usual language barrier between counselors and help seekers. Additionally, migration counselling is still mainly done in one-on-one in-person meetings. The need to schedule appointments, often multiple due to missing documents and complicated structures, leads to long waiting times and inefficiencies for both parties. As a result of regulation and high time requirements, there is little supply of counselling by often overworked experts met by a high demand from newcomers.

[7] https://github.com/digitalfabrik
[8] https://creativecommons.org/licenses/by/4.0

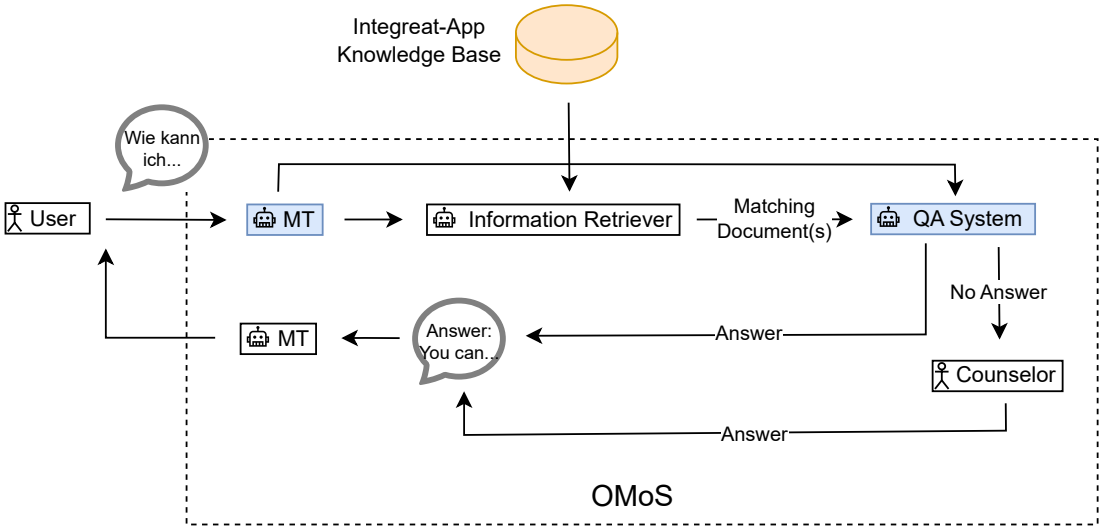

Figure 1.2: The idea of OMoS. The user poses a question to OMoS. The question is first translated using machine translation. Relevant documents are retrieved from the Integreat-App knowledge base and passed to the QA system. If an answer to the question is found, the question is translated back to the user's language and returned to the user. Otherwise, the question is forwarded to a human counselor, whose answers are also translated back to the user's language. The parts of OMoS considered in this thesis are marked in blue: We evaluate different approaches to QA in this context and consider the effect of machine translation on those approaches.

Hence, the idea of an Online Migrationsberatung ohne Sprachbarrieren (OMoS, online migration counseling without language barriers) evolved as an attempt to chain digital possibilities together with classical human migration counseling. The concept of OMoS is to develop a chat-like solution combining resource-efficient answers from the Integreat-App knowledge base and high-quality human counseling in a digital setting. The concept has been awarded with the award for *KI für das Gemeinwohl* (AI for the common good) by the Civic Innovation Platform (CIP).[9]

More specifically, OMoS first tries to answer user questions using a question answering (QA) system based on the contents of the Integreat-App knowledge base, with a focus on trustworthiness. The purpose of this initial QA system to directly answer questions is two-fold: First, we try to provide help seekers with fast and low-threshold information to their questions. Second, we try to filter out simple easy-to-answer questions in order to allow migration counselors to focus on the more difficult and specific cases. If no answers are found or the user is not satisfied, the user can ask to be seamlessly forwarded to a human counseling expert. The communication with the counselor can take place digitally without the need for appointments and long waiting times. In a third step, an in-person meeting can be arranged. On top of that, we want to remove and overcome the language barrier, for example by leveraging machine translation (MT). The idea of OMoS is shown in Fig. 1.2.

[9] `https://integreat-app.de/ki-fuer-das-gemeinwohl`

1.3 Extractive Dataset Construction and Question Answering

OMoS presents a holistic approach to tackle information poverty and consists of a combination of different approaches and techniques in software engineering and development, NLP, and human counseling. In the current work, we focus primarily on the NLP aspect of QA in this German migration context. From various problems accompanying generative QA, we conclude that generative QA and our goal of presenting trustworthy answers in this highly sensitive context are incompatible. We further discuss the problems of LLMs in Section 3.1. Instead, we focus exclusively on extractive QA, i.e., finding answers in already existing documents of a document collection, as opposed to generating new answer text in a generative approach. To this end, we consider the textual knowledge base of the Integreat-App. We evaluate the performance of different neural technologies for this extractive QA. We conduct additional experiments to evaluate multilingual capabilities of models and research the performance implications of machine translation.

In today's world of machine learning, obtaining high quality data to train and evaluate model performance is crucial. There are various datasets for QA available, such as the popular SQuAD (Rajpurkar et al., 2016), its successors and derivates, SQuAD v2 and GermanQuAD (Rajpurkar et al., 2018; Möller et al., 2021), MS MARCO (Bajaj et al., 2018), or HotpotQA (Yang et al., 2018). However, existing datasets are either general-purpose, generative, or consider specific domains different from ours. Additionally, some datasets, for example GermanQuAD, do not incorporate unanswerable questions, which are necessary to evaluate and train models for actual real-world applications.

Thus, we construct a new high-quality dataset, OMoS-QA, which is specifically tailored to the present scenario of question answering in a German migration context. As we plan to focus on extractive QA, we construct our dataset in an extractive manner by eliciting answers to questions from paired documents. We consider documents from the Integreat-App knowledge base as a basis for our dataset. These documents are considered as trustworthy as they are provided by German authorities. We mostly regard non-factoid questions, i.e., questions that are not answerable using simple facts and its answers are instead subjective and require more context. We leverage automatic question generation (QG) using a LLM and a voluntary crowdsourcing approach to collect human answer annotations. The OMoS-QA dataset and its construction are described in Chapter 4.

Most recent neural models in NLP are based on the transformer architecture. This includes both encoder-only models, such as BERT (Devlin et al., 2019), and large language models. While encoder-only models are only able to transform input to an embedding, LLMs are generative models, i.e., they produce new text. Encoder-only models can be employed for different downstream tasks by attaching a new head or output layer and finetuning on the task. LLMs, on the other hand, exhibit the ability to adjust to various downstream tasks without the need for finetuning. Instead, zero-shot or few-shot prompting can be used to "train" these models. We introduce further details in regard to these model classes and the specific models used in this work in Section 2.1.

We focus on possible approaches for the aforementioned QA system in the multilingual German migration context of OMoS. We use our OMoS-QA dataset to train and evaluate possible approaches. As mentioned before, we aim for trustworthy answers to avoid possibly severe negative consequences of giving wrong answers in this sensitive context and therefore only consider extractive QA. To this end, we evaluate two different techniques: First, we finetune a binary classifier to classify whether document sentences are answers or not. We use the encoder-only model DeBERTa v3 (He et al., 2023) as classifier. Correspondingly, the model is queried separately for every sentence in the paired document to elicit the complete answer. Second, we evaluate generative LLMs for extractive QA. For this, we draw inspiration from Henning et al. (2023), who use LLMs to

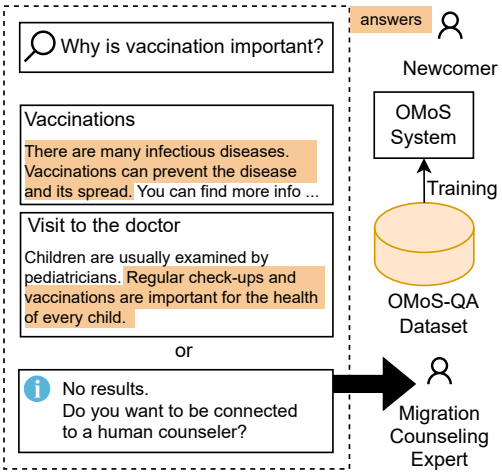

Figure 1.3: Overview of our new OMoS-QA dataset and its usage. The OMoS QA system tries to retrieve relevant documents and extracts answer sentences to the newcomer's question. Training and evaluation of the system are performed on OMoS-QA.

generate answer sentence indices for the exact same purpose, however, on different domain questions on WikiHow articles. We consider different open-weight models by Mistral AI (Mistral-7B (Jiang et al., 2023), Mixtral-8x7B (Jiang et al., 2024)) and Meta (Llama-3-8B, Llama-3-70B (Touvron et al., 2023b)), and compare them to closed-source GPT-3.5-Turbo (Ouyang et al., 2022). We conduct these extractive QA experiments in different settings, languages, as well as zero and few-shot prompting setups. We thereby evaluate model performance on low-resource languages, such as Arabic or Ukrainian, and investigate the performance implications of leveraging machine translation. Furthermore, we conduct a pilot experiment on cross-language prompts, i.e., prompting a LLM with questions in different languages than the document. Lastly, we investigate model performance on detecting unanswerable questions with possible applications in further improving the trustworthiness of our QA system's answers.

Plan of the Thesis. We first give an overview of the technical background in Chapter 2. Covered topics include language modeling in general and the used models in specific, information retrieval and question answering, and the metrics used later on to measure inter-annotator agreement and model performance. We discuss work related to our task in Chapter 3. Specifically, we consider the applications of LLMs in general and go into the chances and risks of generative NLP approaches. We take a look at the construction of other datasets for QA applications and present related work regarding QA and information retrieval. We then describe our modular dataset construction approach in Chapter 4. The construction steps include automatic question generation, human answer annotating, dataset filtering, translating the dataset, and splitting the dataset into train, development, and test partitions. Furthermore, we discuss the development of our custom annotation tool and required prerequisites for leveraging voluntary crowdsourcing for dataset construction. Chapter 5 describes the modeling for our various experiments. We discuss the setup for both the sentence extraction of answer sentences and the detection of unanswerable questions. We consider both LLMs and finetuned classifiers for each experiment setup. Our experiments and the results are described in Chapter 6. We first compare different LLMs in zero-shot and 5-shot settings with DeBERTa and human agreement for German and English OMoS-QA, followed by an evaluation of

models on OMoS-QA translated to additional languages. These results are compared with applying machine translation before prompting the model. In addition, we conduct pilot experiments with cross-language QA pairs and explicit unanswerability detection. Finally, we discuss our work and give an outlook on future research and applications (Chapter 7) before concluding the thesis (Chapter 8).

1.4 Contributions

The main contributions of this thesis are as follows:

OMoS-QA Dataset. We construct and publish OMoS-QA, our new high-quality QA dataset with questions in German and English. We pair automatically generated questions with relevant and trustworthy documents from the knowledge base of the Integreat-App, provided by three municipalities from Southern Germany. The dataset covers a variety of legal, economic, and social topics regarding immigration and every-day life in Germany. The 906 questions are automatically generated with an open-weight LLM using two different prompting methods to elicit diverse questions. The dataset is extractive and includes unanswerable questions, with answers given as lists of answer sentences indices. We use a voluntary crowdsourcing approach to manually create answer annotations. In order to guarantee a high quality of the dataset, we require two annotations per question and only keep questions with high inter-annotator agreement. Since deciding on answer sentences seems to be challenging for non-factoid questions, we propose a novel approach to expand agreement between annotators. We argue that both generative LLMs and crowdsourcing can greatly facilitate the construction of QA datasets and discuss required prerequisites for this. We present the dataset, its modular construction approach, and detailed corpus and agreement statistics in Chapter 4.

Custom Annotation Tool. We develop a novel annotation tool specifically tailored to our use case of providing manual answer annotations. In order to make the bottleneck of our dataset construction, human answer annotations, as efficient as possible, we develop an easy-to-use web-based annotation tool. The annotation tool is completely open-source and available on GitHub [10] under the permissive MIT license.[11] We describe the architecture and design decisions, used technologies, and the features of our annotation tool in Section 4.2.2.

Multilingual Extractive QA Experiments. We conduct various experiments to assess the capabilities of different language models and approaches to extractive QA. Our results show that recent open-weight LLMs are competitive and even outperform GPT-3.5-Turbo on our OMoS-QA dataset. Overall, the used models exhibit high precision and medium recall on the task of extracting answer sentences and high recall in detecting unanswerable questions. We compare our extractive QA setup with generative LLMs to a classification approach using finetuned DeBERTa and find that the latter shows promising performance with a superior F1-score but lower precision, contradicting our goal of presenting only trustworthy answers to our users. Due to the multilingual nature of OMoS, we investigate the performance of LLMs if prompted in Arabic, French, and Ukrainian, which are highly relevant in the migration context. We also study the effect of machine translation in comparison. Leveraging machine translation before prompting the model produces better

[10] `https://github.com/digitalfabrik/integreat-qa-dataset`
[11] `https://mit-license.org`

results than directly querying the model in the investigated languages. Furthermore, we conduct pilot experiments to assess the cross-lingual capabilities of LLMs and find that providing models with document and languages in different languages does not necessarily lead to a deterioration of performance, and depending on the used languages, even provides better results. Our experiments are described in Chapter 6.

Publication. The results of this work are also described in the paper *OMoS-QA: A Dataset for Cross-Lingual Extractive Question Answering in a German Migration Context* (Kleinle et al., 2024). It has been accepted to KONVENS 2024. The dataset collection, the development of the custom annotation tool, and all the experiments were conducted by the author of this thesis.

2. Background

This chapter introduces several theoretical background concepts. First, we establish language modeling and the transformer architecture in general and the models used in this work in specific (Section 2.1). We then introduce question answering and information retrieval (Section 2.2). We conclude the chapter with a summary of the metrics applied to measure inter-annotator agreement in the dataset construction and to evaluate and compare the results of our models (Section 2.3).

2.1 Language Models

According to Jurafsky and Martin (2023), *language models* (LMs) are models that "assign probabilities to sequences of words," i.e., predict (the probability of) the next word from a sequence of previous words. Early attempts to language modeling include statistical approaches such as *n-grams* as well as machine learning techniques, for example *recurrent neural networks* (RNNs). The latter make use of recurrent connections and hidden states to model history and context. However, RNNs (and n-grams) fail to handle distant contextual relations and lack a selective and context-sensitive weighting mechanism of different tokens in the context. Due to the sequential processing of sequences, these models are furthermore hard to parallelize, slowing down the training and processing (Jurafsky and Martin, 2023).

2.1.1 Transformers

The advent of the *transformer* architecture addresses these significant shortcomings with a novel (self-)attention mechanism (Vaswani et al., 2017). Instead of processing words sequentially, transformers attend to the whole input at once and can detect dependencies and context among the whole input sequence. In comparison to recurrent neural networks, where only the last hidden state is available in the next step, the parallelized attention mechanism allows the inclusion of all hidden states. As a result, transformers excel in detecting and modeling distant and context-sensitive dependencies between different parts of the sequence. The size of the input is limited by the so-called *context window*, which describes the maximum input length that can be processed at the same time. Longer input sequences have to be truncated or split and processed in multiple runs. This comes at the cost of requiring a much larger number of model parameters, which ultimately leads to LLMs (Section 2.1.3) with billions or even trillions of parameters (Olariu et al., 2023). Language models employing the transformer architecture are composed of multiple blocks, each containing self-attention and feed-forward layers (Vaswani et al., 2017).

Encoder and Decoder. The original transformer proposed by Vaswani et al. for machine translation is split into an ENCODER and a DECODER. In this architecture, input is first handed to the ENCODER, which generates a contextualized representation of the input, a so-called *embedding*. The model tries to represent the inherent meaning and dependencies of the input in these embeddings. In a second step, the contextualized representation of the input is then passed on and processed by the DECODER to generate an output target sequence (Jurafsky and Martin, 2023). While most current models are employing the transformer architecture in some way, few are using the original encoder-decoder setup. Instead, models are usually *encoder-only* or *decoder-only*. Encoder-only models only encode the input into embeddings and are not able to generate new text. They are

frequently used for text understanding and classification purposes. Decoder-only architectures, on the other hand, take a (possibly encoded) sequence and generate new text. These models are called *generative*.

2.1.2 Encoder-Only Models

In the following section, we introduce two encoder-only models: We first give an overview over BERT, which has been one of the first models using the transformer architecture and paved the way for subsequent similar models (Section 2.1.2.1). We then discuss DeBERTa and its different versions in Section 2.1.2.2. The DeBERTa model family improves the original BERT model and is used in our experiments.

2.1.2.1 BERT

The transformer architecture has gained traction with BERT (Bidirectional Encoder Representations from Transformers; Devlin et al., 2019). The self-attention mechanism utilized by BERT allows for increased contextual understanding compared to previous unidirectional methods by incorporating context from succeeding tokens. Since training models from scratch becomes increasingly challenging and costly with growing model size, BERT—with for the time staggering 110 million parameters in the base version—has popularized the concepts of *pretraining* and *finetuning*. Pretraining describes "the process of learning [...] meaning [...] by processing large amounts of text"(Jurafsky and Martin, 2023). The resulting pretrained model can then be *finetuned* on different downstream tasks with a smaller set of training data—without the need to train from scratch and, as a consequence, substantially faster and cheaper. BERT shows remarkable performance in text understanding and classification applications while lacking the ability to generate new text due to its encoder-only architecture (Devlin et al., 2019). By attaching different *heads* on the pretrained base model and finetuning the head on the downstream task at hand, BERT can be leveraged for various NLP applications.

2.1.2.2 DeBERTa

Due to its strong performance as well as its open-source availability, BERT has seen wide adoption among various NLP applications. It has been developed further and enhanced using different approaches, for example with RoBERTa (Robustly optimized BERT approach; Liu et al., 2019) and DeBERTa (Decoding-enhanced BERT with disentangled attention; He et al., 2021).

DeBERTa and its successors DeBERTa v2 and DeBERTa v3 (He et al., 2023) are still widely used and ranking high on leaderboards such as GLUE (Wang et al., 2018). We finetune and run all our classification experiments solely on DeBERTa v3 large with 304M backbone and 131M embedding parameters and a context window of 1024 tokens.[1] Since we only use DeBERTa in the v3 large version, we henceforth abbreviate this variant as DeBERTa.

2.1.3 Large Language Models

Contrary to encoder-only models such as BERT, state-of-the-art LLMs employ a *decoder-only* approach. This allows LLMs to be *generative* and produce new text and not just embeddings.

[1] `https://huggingface.co/microsoft/deberta-v3-large`

Decoder-only models have been popularized with the GPT series with a breakthrough with GPT-3 with 175 billion parameters (Brown et al., 2020). Due to the model size of LLMs ranging from multiple billions to trillions of parameters, finetuning is not feasible anymore or at least very costly (Olariu et al., 2023). As a result, the standard approach to interact with LLMs is by using *prompts* and *prompt engineering* to "train" the model on the desired task. A common pattern is few-shot—in contrast to zero-shot or one-shot—prompting, which refers to providing examples of how the model is expected to process user input in the prompt (Brown et al., 2020).

A lot of recent LLMs provide different model sizes and instruction-finetuned versions. Instruction finetuning refers to training the model on input-output pairs similar to a chat-like setting. The model therefore learns to follow natural language instructions and to provide output in the expected way.

This section introduces all LLMs that are used in our dataset construction or evaluated in the experiments, starting with Mistral-7B (Section 2.1.3.1) and Mixtral-8x7B (Section 2.1.3.2) by Mistral AI. Subsequently, we give an overview over the latest Llama model family in general and Llama-3-8B and Llama-3-70B in specific (Section 2.1.3.3). We conclude the section by describing the closed-source GPT-3.5-Turbo in Section 2.1.3.4.

2.1.3.1 Mistral-7B

Mistral-7B is a LLM with 7 billion parameters developed by Mistral AI (Jiang et al., 2023). We use the instruction-finetuned version Mistral-7B-Instruct in version v0.2 (Jiang et al., 2023) with a 32k context window.[2] The model is published under the fully permissive Apache 2.0 license.[3] We henceforth abbreviate the model Mistral-7B-Instruct-v0.2 as Mistral-7B.

2.1.3.2 Mixtral-8x7B

Mixtral is a family of LLMs developed by Mistral AI available in 8x7B and 8x22B versions. Mixtral models are built using a so-called "Sparse Mixture of Experts" (SMoE) architecture where the *8x* denotes the number of *experts* and *7B* and *22B* the number of parameters per expert (Jiang et al., 2024). This architecture makes use of a SMoE layer, where input vectors are only processed by 2 out of the 8 expert feedforward blocks chosen by a router. The model output is the weighted sum of the experts outputs. Compared to classical dense LLMs, pretraining and inference is faster since only a limited subset of all parameters are active at a time, e.g., for Mixtral-8x7B with 47B parameters only 13B are active simultaneously (Jiang et al., 2024).

We use the instruction-finetuned version Mixtral-8x7B-Instruct (v0.1) for both question generation (see Section 4.1) and our text extraction experiments.[4] Analogous to Mistral-7B introduced in the previous section, it has a 32k token context window and is published under the Apache 2.0 license. We henceforth abbreviate this model version as Mixtral-8x7B.

2.1.3.3 Llama-3-8B and Llama-3-70B

Llama-3 is the third and latest iteration of Meta's open-weight Llama model family (Touvron et al., 2023a,b). It comes in sizes of 8B and 70B parameters each available in a pretrained and an

[2] `https://huggingface.co/mistralai/Mistral-7B-Instruct-v0.2`
[3] `https://spdx.org/licenses/Apache-2.0.html`
[4] `https://huggingface.co/mistralai/Mixtral-8x7B-Instruct-v0.1`

instruction tuned variant. We use both sizes in the instruction tuned variant for our experiments and henceforth omit the "-Instruct" suffix in the model names.[5] [6] The models are published under a new Lama-3 community license allowing all of our use cases.[7]

2.1.3.4 GPT-3.5-Turbo

For comparison, we include results of the closed-source GPT-3.5-Turbo-0125 (GPT-3.5-Turbo) by OpenAI.[8] GPT-3.5-Turbo is part of the GPT series (Generative Pre-Trained Transformer; Radford et al., 2018, 2019; Brown et al., 2020; Ouyang et al., 2022) and accessible using ChatGPT.[9] We used the paid OpenAI API for our experiments.[10]

2.2 Question Answering and Information Retrieval

Question answering (QA), i.e., answering questions to fulfil users' information needs with neural models, can be approached either *retrieval-based* or *knowledge-based*. While the latter employs the model's internal knowledge, mostly acquired during pretraining, the former is depending on an external knowledge base to answer questions. To this end, retrieval-based QA systems employ information retrieval techniques, i.e., they retrieve content based on user queries. In our highly sensitive and domain-specific OMoS setting, we pursue a purely retrieval-based approach to avoid common problems of generative approaches, such as hallucinations and toxic language (Shah and Bender, 2024; Dahl et al., 2024).

In the following sections, we first introduce the traditional IR pipeline and possible additions (Section 2.2.1). We then discuss the possible approaches to IR (Section 2.2.2).

2.2.1 Information Retrieval Pipeline

An IR pipeline traditionally consists of two stages for a separation of concerns: The RETRIEVER is responsible for finding and retrieving (potentially) relevant documents in the document collection. In doing so, it needs to uphold a good balance between retrieving too many and too little documents, i.e., to strike a balance between retrieving irrelevant and missing relevant documents. At the same time, keeping computing complexity low gets more and more important with the size of the document collection.

The READER subsequently processes the retrieved documents to create an answer to the user's question. Hence, the reader has to decide which retrieved documents actually contain a (partial) answer and extract and/or process the relevant evidence. This can be done either in an *extractive* or *generative* manner. While generative approaches generate new texts, extractive solutions select text from the provided evidence or document to answer the question. It is desirable that the READER creates an understandable and precise answer that actually answers the question.

More recent IR systems often add an intermediate RERANKER component serving as an intermediate step between RETRIEVER and READER. A RERANKER is employed to further distill the candidate

[5] https://huggingface.co/meta-llama/Meta-Llama-3-8B-Instruct
[6] https://huggingface.co/meta-llama/Meta-Llama-3-70B-Instruct
[7] https://github.com/meta-llama/llama3/blob/main/LICENSE
[8] https://platform.openai.com/docs/models/gpt-3-5-turbo
[9] https://chatgpt.com
[10] https://platform.openai.com/docs/api-reference/chat/create

documents with a focus on high precision and quality over quantity of the output. The RERANKER is also responsible for sorting or ranking the results according to their relevance. However, task ranges between these three components are fluid.

Another possible addition to this architecture is a so-called QUERY REWRITER to preprocess user queries and questions to improve the results of the whole IR pipeline. It aims to correct misspellings, reduce ambiguity, and increase the precision of the queries.

2.2.2 Information Retrieval Approaches

Prior to the rise of neural networks and LLMs, term-based approaches such as term frequency-inverse document frequency (tf-idf) functions and especially (Okapi) BM25 (Robertson et al., 1994) have been prevalent for IR. With the rise of neural networks and especially LLMs these statistical methods are increasingly superseded by neural approaches (Zhu et al., 2024). Initial neural models like BERT (Devlin et al., 2019) are known to have problems with "short and ambiguous" queries as well as with "lengthy content and substantial noise" in documents. LLMs can mitigate these challenges better through their superior language understanding and reasoning capabilities (Zhu et al., 2024). Current state-of-the-art LLMs offer "superior semantic capability and excel at understanding complicated user intent." Zhu et al. (2024) illustrate different approaches to leverage LLMs for IR and QA in all four aforementioned pipeline stages.

However, the use of LLMs comes with several drawbacks and risks: Generative language models suffer from hallucination, wrongly cite evidence and spread misinformation (Henning et al., 2023). LLMs are known to make use of toxic or discriminating language and to follow or even amplify biases and stereotypes present in training data (Shah and Bender, 2024). Additionally, using and especially training LLMs is costly in terms of hardware, power consumption and time.

2.3 Metrics

We first introduce the Jaccard index, also known as intersection over union (IoU) (Section 2.3.1). Subsequently, we present the common metrics precision, recall, and F1-score (Section 2.3.2).

2.3.1 Jaccard Index

As a measure for computing agreement we use a chance-corrected version of the Jaccard index. For each double annotated question q_i we have two sets of selected answer sentences $A_{i_a} \subseteq S_i$ and $A_{i_b} \subseteq S_i$, where S_i is the set of all sentences of the document d_i, and a and b index the two annotators. The Jaccard index and therefore the observed agreement agr_{obs} for a question q_i and a document d_i are then defined as follows:

$$agr_{obs} = J(A_{i_a}, A_{i_b}) = \frac{|A_{i_a} \cap A_{i_b}|}{|A_{i_a} \cup A_{i_b}|} \tag{2.1}$$

For $A_{i_a} = A_{i_b} = \emptyset$ we set $J(A_{i_a}, A_{i_b}) = 1$ as both annotators completely agree that there is no answer.

Chance Correction. In order to account for the possibility of authors just agreeing "by chance," chance correction can be applied (Opitz, 2024). We assume over-simplifying that the prior probability $P(sel)$ of selecting a sentence $s_{i_k} \in S_i$ is independent of the question, document, and annotator, and compute it as the total fraction of sentence selections over two times the total amount of sentences in the corpus (as each document receives two annotations):

$$P(sel) = \frac{\sum_{i=1}^{n}(|A_{i_a}| + |A_{i_b}|)}{2 * \sum_{i=1}^{n}|S_i|} \tag{2.2}$$

The expected probability $agr_{exp} = P(agr)$ that two random annotations agree on a sentence being an answer is then:

$$agr_{exp} = P(sel)^2 \tag{2.3}$$

In our case, $P(sel)$ is 0.1856 and the expected agreement agr_{exp} amounts to a Jaccard index of only 0.0344 and therefore agreement of random annotations is unlikely. As a result, chance correction does not have a big influence on the IAA as the results only differ slightly.

The chance-corrected Jaccard index can then be calculated as follows:

$$J_{cc}(A_{i_a}, A_{i_b}) = \frac{agr_{obs} - agr_{exp}}{1 - agr_{exp}} \tag{2.4}$$

2.3.2 Precision, Recall, and F1-Score

As evaluation metrics, we use *precision* (P), *recall* (R), and *F1-score* (F). Precision and recall are defined as follows:

$$P = \frac{|S_{retrieved} \cap S_{relevant}|}{|S_{retrieved}|} \tag{2.5}$$

$$R = \frac{|S_{retrieved} \cap S_{relevant}|}{|S_{relevant}|} \tag{2.6}$$

In other words, precision measures how many retrieved elements are actually relevant while recall describes the fraction of relevant elements that are retrieved.

The F1-score is defined as the harmonic mean of recall and precision and thus captures both metrics (Christen et al., 2023). The F1-score is a simplified version of the more general F_β-measure, which allows to adjust the relative importance of precision and recall. For the F1-score they are given the same importance and the formula is as follows:

$$F1 = \frac{2}{P^{-1} + R^{-1}} = 2\frac{PR}{P + R} \tag{2.7}$$

According to Bertels et al. (2019), "approximate each other relatively and absolutely." As further shown in Appendix A, the F1-score and Jaccard index are directly related:

$$J(A, B) = \frac{F1(A, B)}{2 - F1(A, B)} \leq F1(A, B) \tag{2.8}$$

The Jaccard index therefore punishes instances with low agreement harder than the F1-score.

3. Related Work

In this chapter, we present and discuss work related to our task. We first show the applications, capabilities, and risks of LLMs, and thus motivate the focus on extractive QA in our highly sensitive application setting (Section 3.1). We then compare different existing QA datasets, show their limitations, and reason why they are unsuitable for our setting (Section 3.2). Subsequently, we introduce several approaches to modeling extractive QA in Section 3.3.

3.1 LLM Applications and Risks

The research in NLP has produced astonishing advances in the last few years in various areas. Since a breakthrough with GPT-3 (Brown et al., 2020), interest in and usage of LLMs has rapidly increased. The emergent capabilities of recent state-of-the-art large language models (LLMs) has led to the application of LLM-based approaches in nearly every NLP field and task. Extractive and encoder-only question answering is increasingly replaced by newer generative approaches with LLMs such as GPT or Llama (Ouyang et al., 2022; Touvron et al., 2023b; Zhu et al., 2024). LLMs have found applications in finance (Li et al., 2023), medicine (Thirunavukarasu et al., 2023), and software engineering (OpenAI et al., 2024; Chen et al., 2021), among others.

Capabilities. Current state-of-the-art LLMs offer "superior semantic capability" and "excel at understanding complicated user intent" (Zhu et al., 2024). Models have remarkable capabilities in generating qualitative texts (Clark et al., 2021) and code (Mcnutt et al., 2023; OpenAI et al., 2024; Chen et al., 2021). Researches have shown the flexibility and generalization of LLMs to unfamiliar downstream tasks (Zhu et al., 2024; Ouyang et al., 2022). Some researches already claim superhuman performance of LLMs on some NLP tasks (Wang et al., 2019a) or even see them as a step to general artificial intelligence (y Arcas, 2022; Bubeck et al., 2023).

Criticism and Risks. However, for example Tedeschi et al. (2023) question whether this is justified and detect certain biases favoring machines over humans in popular NLP benchmarks. Furthermore, even though LLMs are applied to more and more use cases and have taken a prevalent position in today's NLP applications, their use comes with several drawbacks and risks. First and foremost, LLMs are costly to train, finetune, and even to just operate for inference due to their sheer size with billions or even trillions of parameters. Apart from the hardware costs of GPUs and other processing units, which are required to have sufficient memory to fit the models, energy consumption and network bandwidth are crucial cost factors as well. These demands are not only a monetary and time-wise issue, they additionally leave a considerable environmental footprint. Furthermore, these factors prevent the democratization of LLMs as only some entities have the resources to train and develop new models.

Even more relevant for our present application, LLMs also exhibit problematic behavior apart from any material issues. Shah and Bender (2024) compare employing LLMs and other generative approaches for information access and question answering to classical *discriminative* IR, which retrieves and ranks existing content. They list a number of potential risks and harms: First, they discuss *ungrounded answers*, i.e., incorrect responses, that can lead to the spread of misinformation, and, depending on the use case, adverse to disastrous consequences. Another problem of neural

networks in general and LLMs in specific identified is the replication and even amplification of *biases* present in the training data. These biases are ranging from general over- or underrepresentation to sexist, racist, or other toxic behavior. Shah and Bender make a case that generative models increase problems with biases and toxic behavior by replicating biases without any original context. Additionally, they name a *lack of transparency* as shortcomings of LLMs as, at the moment, it is not understood why some output is produced and on what training data it is based. Without additional information on the source and reasoning behind produced text, it is impossible to put the answers into context and assess their correctness. They also question the ethicality due to labor exploitation for data labeling, neglect of privacy rights and copyright to acquire training data, and environmental costs as a whole. Due to all previously mentioned shortcomings, Shah and Bender classify the use of LLMs in sensitive applications such as in court, for therapy, or in medical environments as "detrimental" and raise questions about trust and trustworthiness. These apparent shortcomings of LLMs and generative approaches in general lead Shah and Bender to question the application of these techniques for IR completely.

The results of Dahl et al. (2024) back this critical view of generative techniques for QA and IR tasks. They have researched the performance of LLMs on legal cases in the United States. In doing so, they have found LLMs to hallucinate in more than half of all presented cases for verifiable questions. GPT-4 performs best with a hallucination rate of "only" 58%, while e.g., Llama-2 hallucinates in close to 90% of all cases. We expect similar results for generative approaches for QA in a German migration context, as information is short-lived, highly person-dependent, and often very nuanced.

3.2 Datasets

Data and datasets are the foundation for all machine learning applications. Data is required for pretraining, finetuning, and evaluation of models and therefore indispensable. While acquiring enough datapoints to allow for an effective pretraining or finetuning and a meaningful evaluation is necessary, high quality is even more important. Chandrasekhar et al. (2023) show that models trained on smaller but high quality datasets outperform those trained on big but unspecific or automatically constructed datasets. In this section, we first introduce several common and popular QA datasets (Section 3.2.1). We argue that none of the existing datasets are applicable for our purposes and evaluate existing literature on dataset construction in order to be able to construct a dataset specifically tailored to trustworthy QA in a German migration context (Section 3.2.2).

3.2.1 Existing QA Datasets

To this end, various datasets are available. One of the most popular datasets for QA, SQuAD (Stanford Question Answering Dataset; Rajpurkar et al., 2016), consists of more than 100,000 open-domain QA pairs. SQuAD is an extractive QA dataset based on Wikipedia articles and focuses solely on factoid questions. Questions are collected and answer spans are marked using crowdsourcing. SQuAD v2 extends the original SQuAD dataset with more than 50,000 unanswerable questions. Derivates of SQuAD have been created for other languages, such as for German (Möller et al., 2021), French (d'Hoffschmidt et al., 2020), or Korean (Lim et al., 2019). HotpotQA (Yang et al., 2018) is also based on Wikipedia articles. In contrast, answering questions from HotpotQA requires understanding and combining information from multiple articles. The dataset includes more than 100,000 questions accompanied by supporting sentence-level evidence required for reasoning about the answer. Questions are also factoid. MS MARCO (Bajaj et al., 2018) is another large dataset for natural language understanding and reading comprehension, sourced from

over 1 million questions to the Bing search engine. Answers are written by humans specifically for this dataset and are therefore not extractive. However, excerpts from documents are included, which provide context required to be able to answer the questions.

However, these and other existing datasets are not suitable to our application in a German migration context. The datasets are either not extractive, focus on factoid questions, lack unanswerable questions, or are open-domain or specific to domains different from ours.

3.2.2 QA Dataset Construction

We consider the steps to the construction of a QA dataset separately: We first study related work on question generation and shortly discuss question rewriting. Subsequently, we cater to different approaches to (human) answer annotation.

Question Generation. In general, question generation (QG) has been shown to produce good results using automated approaches (Han et al., 2022; Bechet et al., 2022). Kumar et al. (2019) propose an approach to cross-lingual question generation to collect questions in further languages.

Research of Dugan et al. (2022) shows that using summaries instead of full paragraphs vastly increases the performance of QG and reduces the risk of irrelevant or inept questions for answer-agnostic question generation. While this holds especially true for human summaries, they show a similar, albeit less strong result for automatic summaries.

Yuan et al. (2023) further argue that for longer contexts and non-factoid open-ended answers, questions are better "posed about abstract ideas rather than simple context paraphrasing." Furthermore, they provide valuable insights in prompt-based question selection methods to choose the most suitable question from a set of candidate questions and propose an *averaged prompt-based score* (APT) for the aforementioned non-factoid question setting.

Henning et al. (2023) report that writing diverse high-quality questions is difficult for humans knowing the corresponding text, especially if unanswerable questions are of interest. In order to circumvent that challenge, human crowd workers are only shown title, first paragraph, and keywords generated from the document. Annotators are then asked to create six questions with varying question words that might be answerable with the full document. They require two answer annotations per question and a sufficient inter-annotator agreement, as the lack thereof indicates "ill-posed or too close to the overall topic" questions. The researches consider QA on WikiHow articles on various topics.

Question Rewriting. Brabant et al. (2022) propose question rewriting to adjust in-context, e.g., "What are the emergency numbers provided in the text?," to out-of-context questions, e.g., "What emergency numbers are available?." They claim that this improves performance of QA systems, as it allows understanding of conversational questions, i.e., questions referring to previous conversation turns. This question rewriting can also be applied to automatically generated questions, as LLMs might not always create out-of-context questions if provided with the full-text prompt.

Annotations. Contrary to automatic question generation, answer annotations are mainly done manually by humans for most QA datasets. Crowdsourcing, usually with paid crowd workers, is employed for annotations for most larger datasets (Rajpurkar et al., 2016; Bajaj et al., 2018; Yang et al., 2018). While some researchers have shown competitive results of models trained on synthetic

data (Alberti et al., 2019; Puri et al., 2020; Bartolo et al., 2021), in general, human generated data and annotations seem to be superior for most NLP applications (e.g., Chandrasekhar et al., 2023; Pradhan and Kuebler, 2022; He et al., 2015). In order to assure high trustworthiness of our answers and correct detection of unanswerable questions, given a paired context document, this is particular important to us (cf. Rajpurkar et al., 2018; Liu et al., 2019).

Henning et al. (2023) conduct answer annotations on a sentence level, i.e., sentences that answer the question or are part of evidence for an answer, are manually selected. They report a moderate agreement on these answer annotations. For their gold-standard dataset, agreement is calculated per question on the selected sentences of the two annotations. They use a F1-score for the inter-annotator agreement of 0.3 as threshold to filter questions. For the ground-truth answers, Henning et al. use the union of selected sentences, as disagreements are due to "how much context to include." They have annotated 570 questions from 95 documents.

3.3 Modeling and Extractive Question Answering

Question answering can be conducted in both an extractive and a generative manner. Extractive approaches try to answer a question only with a span, sentence, or paragraph directly extracted from a document with evidence. Generative variants, on the other hand, generate their answers from knowledge obtained during (pre)training. Due to LLMs, which show superior textual understanding and reasoning, generative QA is becoming increasingly popular. Combinations of extractive and generative approaches, for example in the form of retrieval-augmented generation (RAG), are possible. RAG, albeit being of a generative nature, incorporates information extracted from documents provided in the prompt. Luo et al. (2022) provide an extensive comparison of generative and extractive approaches for QA and information retrieval. According to their results, "extractive readers perform better in short context" and show "better out-of-domain generalization." Liu et al. (2024) show significantly worsened performance of LLMs if important information is presented in the middle of the prompt, which is especially a problem for long contexts. Due to these problems and general issues of LLMs discussed in Section 3.1, e.g., hallucinations, we only consider extractive QA in the following section.

Standard extractive QA approaches usually predict token spans, which include the answer or supporting evidence (Seo et al., 2018; Clark and Gardner, 2018). Wang et al. (2019b) extract evidence supporting the answer on a sentence level using a finetuned GPT model (Radford et al., 2018).

Henning et al. (2023) propose a novel extractive QA approach using generative LLMs. They prompt ChatGPT to generate a list of sentence indices of answer sentences in the document in both a zero-shot and a 5-shot setting, e.g., `[1, 4, 7]`. In the few-shot setting five randomly sampled instances where used, consisting of three answerable and two unanswerable questions, each referring to different documents. Sentences in the documents are enumerated using a `[i]` prefix. Henning et al. report precision around 38% and recall of 52.6% in the zero-shot and 42.8 in the 5-shot setting for extracting answer sentences. While ChatGPT fails to identify unanswerable questions in the zero-shot setting (recall of 7.2%), providing examples in a 5-shot setting significantly improves model performance on this task (60.3%).

4. Dataset Construction

In this chapter we describe the creation of the OMoS-QA dataset consisting of over 900 manually annotated QA pairs based on textual content from the Integreat-App. The dataset is intended for extractive QA at the sentence level, hence answers are given as a (possibly empty) list of sentence indices of the corresponding document. We construct our dataset from German and English QA pairs.

For the dataset creation we adopt a two-pronged approach leveraging both automatic question generation and manual answer annotations. In the following sections we specify the various steps of the dataset construction, namely question generation and manual question auditing (Section 4.1), human annotation using a crowdsourcing approach (Section 4.2) as well as filtering for a ground-truth dataset (Section 4.3). Furthermore, we use machine translation to provide the dataset in additional languages (Section 4.4). We address the dataset split in train, development (dev), and test partitions (Section 4.5) and present corpus statistics of the final dataset (Section 4.6). Subsequently, we analyze our initial attempt of manually collecting complete QA pairs through volunteers and discuss possible changes to the used dataset construction process (Section 4.7). Finally, we provide a short conclusion of the dataset creation process and the use of crowdsourcing for dataset creation (Section 4.8). An overview of the dataset construction process and its steps can be seen in Fig. 4.1.

4.1 Question Generation

Instead of collecting manually written questions we use the capabilities of LLMs in natural language understanding and processing to generate both English and German questions for the OMoS-QA dataset.

To facilitate the diversity of the dataset and to include both answerable and unanswerable questions, we employ two different question generation strategies for every document: Question generation with (Section 4.1.2) and without (Section 4.1.3) evidence in the model input. An overview of question generation is shown in We expect that generating questions without evidence produces a substantial amount of unanswerable questions while most questions generated with evidence should be answerable. Unanswerable questions are desirable to simulate the absence of answers to a user's question in the document collection in general and in the retrieved documents in particular. As a consequence, the READER module is required to detect unanswerable questions instead of just outputting wrong answers.

This chapter furthermore describes the preprocessing of the documents (Section 4.1.1) and the postprocessing of the questions (Section 4.1.4).

4.1.1 Document Preprocessing

For both QG strategies we draw on German and English documents from the Integreat-App. Our corpus consists of documents from three municipalities in south Germany.[1] We have retrieved the documents using the Integreat-API[2] on 2024-02-02. All documents, questions, and answer

[1] The city of Munich and the districts (Landkreise) Augsburg and Rems-Murr-Kreis.
[2] `https://digitalfabrik.github.io/integreat-cms/api-docs.html#pages`

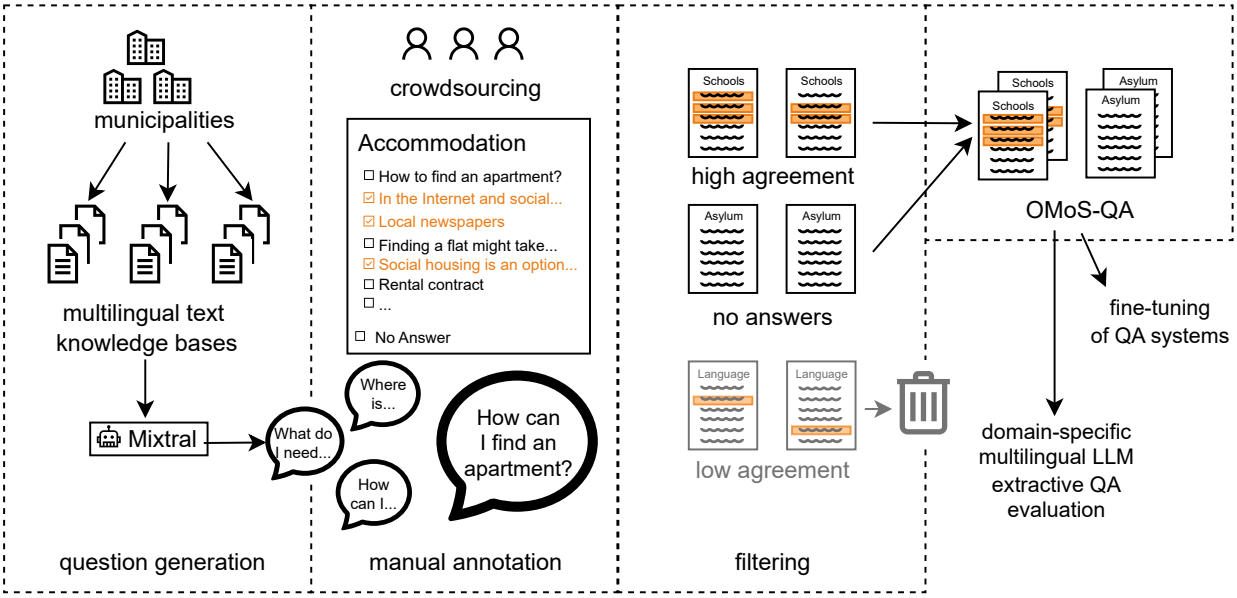

Figure 4.1: The construction of the OMoS-QA dataset. Documents are taken from real-life multilingual knowledge bases. Questions are generated using Mixtral-8x7B, but answers are annotated manually using crowdsourcing. The double-annotated dataset is then filtered on a question-level according to inter-annotator agreement.

sentences therefore resemble the state of the Integreat-App on that date. Subsequent changes to the documents in the Integreat-App are not reflected in the dataset.

The Integreat-API returns the documents in a JSON-format containing various properties, including the plain-text titles and HTML text contents. Common topics in the Integreat-App are relevant for everyday life such as education, work, language, and health or closely related to the migration process, for example questions about the asylum process, financial support or visa applications. Since most municipalities start providing content in the Integreat-App from a generic template created by Tür an Tür Digitalfabrik and usually cover the aforementioned similar topics, we apply filtering to remove duplicated documents in order to avoid overrepresenting individual documents in the final dataset. To this end, documents with duplicated titles are removed.

A substantial part of the documents of all three municipalities cover specific services such as meeting places, support offers, language courses or counseling services. These documents generally include a location, contact information, and opening hours as well as information about costs and accessibility. In order to increase the percentage of documents with more general instructions and information, a keyword-based filtering is applied to remove some documents with specific services. We want to strike a good balance between specific and general information to allow for helpful results in all cases. Documents containing the text "The service is free," present in a lot of texts covering specific services, are removed.

In order to allow for better readability by both humans and LLMs, the HTML tags are stripped, and the actual textual contents are extracted. We use the Python library *Beautiful Soup*[3] for this purpose. Subsequently, the text is adjusted such that all sentences are separated by the newline character `\n` by applying a regular expression (regex). This leads to some incorrect line breaks, e.g., for abbreviations like "i.e." and "z.B." as well as after enumerations, for example "1.". This

[3] `https://beautiful-soup-4.readthedocs.io`

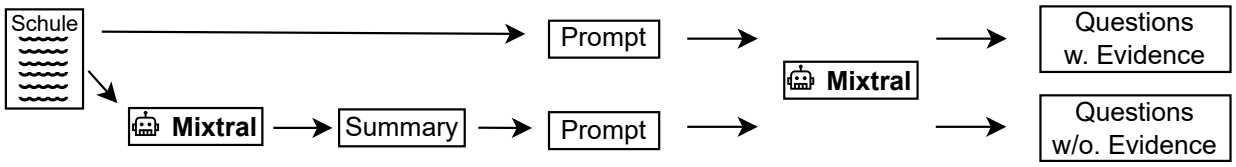

Figure 4.2: Question generation with and without evidence. For question generation with evidence, the model is prompted to generate questions on the full document. For question generation without evidence, we first prompt the model to generate a three word topic summary, for which we then generate questions.

could be improved in future work. Keywords and phone numbers are removed from the document. Lastly, only documents with a minimum length of 500 characters are considered.

In the following, the term *document* always refers to the stripped and newline-separated text prefixed with the title.

4.1.2 Question Generation With Evidence

We use the Mixtral-8x7B LLM to generate questions with evidence on a per-document basis with the aim of eliciting high-quality, diverse, and potential realistic questions for each document. Hence, we prompt the model for three questions at a time to foster the variety of the generated questions by avoiding a potential bias in the first returned questions. Additionally, asking for multiple questions allows for more cost-efficient QG as only one document per three questions is needed in the prompt. In addition to the questions, the prompt also instructs the model to include the corresponding answer sentence indices in the response. These sentence indices would allow us to preselect answers in the following manual annotation process. We request the output in a pattern allowing for simple postprocessing by prefixing questions with `Qi:` and answers with `Ai:` for $i \in \{1, 2, 3\}$.

To avoid complicated, combined, and both too specific and too generic questions, we modify the prompt with additional constraints and include positive and negative examples. For German documents the examples are translated to German while the rest of the query is given in English.

In other words, we query the model using the prompt shown in Fig. 4.3 separately for every document. To this end, the instruction format in Fig. 4.4 is used with the prompt and the full document.

Results. Llama-2 in both the 7B and the 70B version does not produce results of reasonable quality. Responses include lots of other text and made-up conversations between user and assistant. However, the prompt works well using both GPT-3.5-Turbo and Mixtral-8x7B, and the output mostly follows the requested pattern for both models. Mixtral-8x7B sometimes separates sentence indices with a hyphen to indicate continuous answers. Furthermore, the model does not always stick to the three question per document limit.

Because of potential licensing problems and API usage costs with GPT-3.5-Turbo, we choose Mixtral-8x7B and discard questions generated with GPT-3.5-Turbo. The aforementioned minor issues can be worked around with postprocessing.

In future work, this QG prompt could be further optimized to e.g., follow a JSON-format or with a few-shot prompting approach.

```
Give three simple and short one-part questions that can be answered with the
users message. The question should be specific and in easy-to-understand language.
Bad examples:
- What services are offered?
- How many people live in Germany?
Respond by giving the questions and the answers.
For the answers, only give the line numbers, do not give whole sentences.
Good example:
"""
Q1: What language courses are available?
A1: 3, 4, 5
Q2: How can I find language courses?
A2: 7
Q3: What does language level B2 mean?
A3: 6
"""
```

Figure 4.3: Prompt for question generation with evidence. The model is prompted to generate three questions that can be answered with the document and the indices of the corresponding answer sentences.

```
<s> [INST] {prompt} [/INST\nUser: {document}]\nAssistant:
```

Figure 4.4: Instruction format for question generation with Mixtral-8x7B.[4]

4.1.3 Question Generation Without Evidence

To allow for unanswerable questions and a higher question diversity, we harness QG without evidence as second strategy. In other words, we use a prompt not including the full document, such that it is impossible for the model to know which questions are answerable. Following the results of Henning et al. (2023) and Dugan et al. (2022) that QG from summaries is yielding better results and more relevant and correct questions, we adopt a similar approach for OMoS-QA. As manual summarization recommended by Dugan et al. (2022) is too time-consuming for this work, we instead employ Mixtral-8x7B to provide automatic summaries.

While Henning et al. (2023) use title, first paragraph, and generated keywords and Dugan et al. (2022) use a full-text summary for QG, we find that automatically generated three word topics leveraging the following prompt deliver the best results for our purposes. A word count of three gives the most concise topic description without revealing too much of the actual contents to elicit unanswerable questions. Examples for the generated topics are “domestic violence support,” “refugee counseling services,” or “recognition of degrees.” The prompt used to generate the three-word topic summary is shown in Fig. 4.5.

Subsequently, the model is prompted similarly as in Section 4.1.2 with translated examples for German documents. As there is no knowledge of the exact document, no answer lines are requested. We show the prompt for question generation without evidence in Fig. 4.6. Similar to prompting for QG with evidence, the prompt is passed to the model as system prompt and the three word topic as user message with the instruction format shown in Fig. 4.4.

```
Give the topic of the text using max. 3 words.
```

Figure 4.5: Prompt to generate the three word topic summary for question generation without evidence.

```
You are a refugee/newcomer in Germany and are looking for help. Give three simple
and short one-part questions that could be answered by a text with the following
topic. The question should be specific and in easy-to-understand language.
Bad examples:
- What services are offered?
- How many people live in Germany?
- Does the user...?
Good example:
"""
Q1: What language courses are available?
Q2: How can I find language courses?
Q3: What does language level B2 mean?
"""
```

Figure 4.6: Prompt for question generation without evidence. The model is prompted to generate three questions about the three word topic summary of the document.

Results. The results using this query are quite similar in quality to QG with evidence. However, as expected some questions are not answerable using the document. We employ only Mixtral-8x7B because of the aforementioned licensing issues with GPT-3.5-Turbo and unsatisfactory results with Llama-2.

4.1.4 Question Postprocessing and Auditing

All questions are manually filtered superficially and in some cases corrected or enhanced by the author. This includes fixing of typos, removal of unfitting questions, and generalizing questions too close to the document, i.e., questions that are only understandable given the context of the document. An example for the latter is "What are the emergency numbers provided?," which is rephrased to "What emergency numbers are available?." Additional examples can be found in Appendix B.

In total, we collect 1,844 German questions for 548 documents and 3,062 English questions for 652 documents. Around 60% of the questions are generated without evidence from the three-word topic as described in Section 4.1.3 with the rest generated with evidence as specified in Section 4.1.2.

4.2 Human Annotations

Due to high real-world risks of providing incorrect answers, the task of finding ground-truth standard answers within documents—in contrast to automatic QG—resides with human annotators. This section first gives motivation for using human annotations (Section 4.2.1). We then describe the custom annotation tool used (Section 4.2.2) and finally present the results (Section 4.2.3).

4.2.1 Motivation

While poorly phrased, ambiguous, or unfitting questions are to be expected from users, incorrect answers could severely impact training and evaluation of a QA system. Given the nature and context of our work with potentially life-changing impact for a particularly vulnerable group of people, this is extremely important. In order to mitigate this threat and to account for voluntary or involuntary mistakes, biases, and subjective answers by annotators, we resort to multiple human annotations per question and only retain and utilize QA pairs with high inter-annotator agreement.

As a result, human annotations are the bottleneck in the size of the dataset, amplified by the need for at least two annotations per question. Without funding for professional annotators, we resort to voluntary crowdsourcing. The annotators are recruited on a voluntary basis from German NGOs and volunteers in the migration context, employees of Tür an Tür Digitalfabrik, and in the personal environments of the author and the advisors of this thesis.

The annotation task is framed as the selection of one or multiple complete sentences that help to answer the question. Annotators are shown a question together with the sentence-by-sentence selectable document. Any number of sentences can be selected and deselected again, however, if no answer is found in the text, a separate checkbox has to be selected to explicitly confirm this decision. To facilitate the annotation process, we develop a custom web-based annotation tool.

4.2.2 Annotation Tool

This section describes the architecture and components of our custom-built annotation tool. As mentioned beforehand, human annotations are the bottleneck in our dataset construction process to collect enough datapoints for our purposes. As we do not have a budget for paid annotators or experts and want to avoid potential bias by exclusively annotating the dataset ourselves, we opt for a crowdsourcing approach with volunteers. Hence, our main design objective of the annotation tool is ease of use to facilitate and expedite the annotation process. Furthermore, we aim to guide our untrained annotators to prevent misunderstandings and incorrect execution of the task at hand. The annotation tool should produce consistently formatted annotations, which are therefore easy to postprocess. A screenshot of the annotation tool is shown in Fig. 4.7. Additional screenshots can be found in Appendix D.

Architectural and Design Decisions. There are a number of existing annotation tools for various purposes. For text labeling and NLP, commonly used open-source tools include INCEpTION (Klie et al., 2018), Argilla (Daniel and Francisco, 2023) or doccano (Nakayama et al., 2018). Existing tools are mature and customizable, yet they are not applicable for our use case. All considered off-the-shelf annotation tools require user-based authentication. Since we employ an uncontrolled crowdsourcing approach, i.e., we plan to spread the task among various groups hoping for continued circulation by third-parties, creating users for every volunteer is not feasible. In addition, while most tools are customizable, they often focus on more complicated annotation task and are therefore not as easy to use as they could be. Hence, we develop a custom-built annotation tool specifically tailored to our use case of extracting answer sentences for QA.

In order to allow for the most straightforward user experience, we make the following design decisions: First, we employ a web-based frontend-backend architecture, which allows our annotators to easily access the annotation tool without the need to install or download. Instead, annotators simply have to click on a link to start annotating. Second, we avoid any kind of authentication and instead just map users to their annotations with a unique identifier. This greatly reduces the

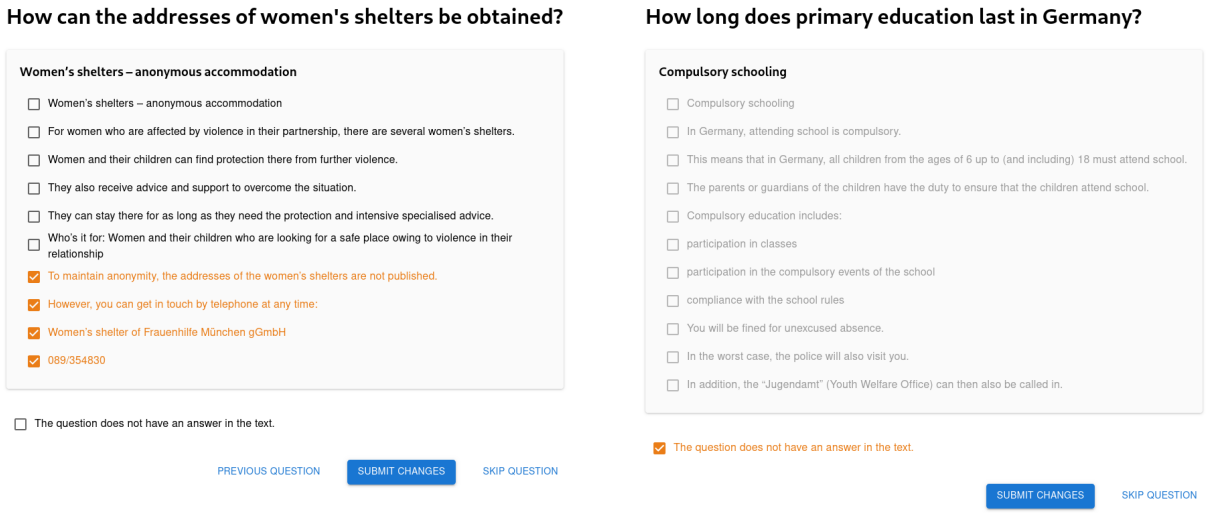

Figure 4.7: Screenshots of the custom annotation tool for human answer annotations.

initial obstacles required to annotate for volunteers. While this potentially allows for abuse, this possibility can be mitigated with subsequent agreement analysis and filtering of the annotations described in Section 4.3. Last, we allow annotators to do as many or as few annotations at a time as they want to. Instead of assigning batches, we select the next question on-the-fly on a per-question basis. Thus, every single annotated question is valuable for us and annotators are not required to do a fixed number of questions, which might be overwhelming with annotators quitting.

For the decisions on languages and frameworks to use, we consider several criteria: Support in the construction of lightweight and user-friendly software, ease of use for developers, and the size of the community and general adoption of the solution. We therefore choose well-established and modern programming languages and frameworks for the respective purposes. We develop a web-based annotation tool consisting of a Ktor[5] backend and a React[6] frontend written in Kotlin[7] and TypeScript[8] respectively.

Backend. The Ktor framework and Kotlin programming language allow us to write a lightweight and concise backend. We create several *GET* and *POST* REST endpoints to fetch annotations, questions, and properties and to save and update user annotations. In order to persist the documents, questions, annotations, and user actions in the backend, we employ a PostgreSQL relational database.[9] We access the database with the Exposed framework.[10] To allow for possible future changes in our requirements, we set up multiple database tables to correctly model the relationships of our entities, i.e., annotators, documents, questions, and annotations. We allow for archiving of questions and annotations and keep track of these changes for possible future analysis. Annotations are for example automatically archived, when a user edits the annotation after initial submission. Comments and skipped questions are also persisted. Questions and documents can be imported from and exported to JSON files by calling the Clikt command line interface.[11]

[5] `https://ktor.io`
[6] `https://react.dev`
[7] `https://kotlinlang.org`
[8] `https://typescriptlang.org`
[9] `https://postgresql.org`
[10] `https://jetbrains.github.io/exposed`
[11] `https://ajalt.github.io/clikt`

The algorithm to decide on the next question for a user is simple and mostly based on a random selection. We try to efficiently assign questions to avoid both unusable questions—due to too little annotations—and redundant annotations, i.e., more than the required two per question. Hence, we attempt to randomly assign questions to users that are already annotated exactly once. However, we try to minimize the chances of another user receiving this question by employing a threshold of 10 questions with exactly one annotation. If we fall below this threshold, we randomly assign a question without any previous annotations. Archived and skipped annotations are excluded from the threshold. Additionally, a user is never asked the same question twice, i.e., if a question is skipped or annotated, it will not be shown again to the same user. The Kotlin code to decide on the next question is shown in Fig. D.2 in Appendix D.

Frontend. In the web-based frontend we leverage the popular React library for web interfaces and use TypeScript, a type-safe language based on JavaScript. For our user interface, we mostly follow the Material Design guidelines, an open-source design system developed by Google,[12] by using the Material UI component library for our UI components.[13] We implement a landing page giving instructions on the annotation task and providing background information for the volunteers. Users are required to give consent to the anonymous processing, publication, and usage of their annotations for machine learning before starting to annotate. The user interface is available in English and German. It is possible to select the language of the questions to annotate, as a default, however, it is set to random.

The annotation page itself is kept simple, only displaying the question, instructions, and individually selectable sentences of the document. In addition, a separate checkbox for unanswerable questions, a free-text comment field, and buttons to submit, skip, and show previous questions are shown. Annotation history is only available to the user per session. Thus, it is not possible to edit a previous annotation in the next session. Screenshots are included in Appendix D.

To anonymously connect the annotations to users, a universally unique identifier (UUID) is created and saved in the browser's local storage upon first access of the annotation tool. While this user tracking might be sidestepped by users deleting the local storage or using multiple browsers or devices, this is deemed sufficient for the present use case of mapping annotations to users and preventing the same question from being shown multiple times. In future work, this UUID can be used to do a per-user filtering on the gold-standard answers by removing annotations from users with low overall agreement.

Additional but in the end unused features of the annotation tool include preselecting answer sentences based on previous annotations or model suggestions and being able to select the question source, e.g., questions from the city of Munich. Furthermore, the tool includes a page to compare the annotations of different annotators with a side-by-side view of the respectively selected sentences. This comparison view allows selection of different agreement categories, e.g., questions with full agreement, an overlap of selected sentences, or an agreement of zero.

4.2.3 Annotation Results

We have gathered 3,688 annotations for 1,944 questions in total by 238 annotators. These annotations amount to 1,744 questions with two annotations (German: 1,268, English: 476) for 863 different documents. 46 annotations have been revisited and changed after initial submission. Users

[12] `https://m3.material.io`
[13] `https://mui.com/material-ui`

annotated a little over 15 questions on average with a mean Jaccard agreement index of 0.31 per user.

In future work this annotation process could be continued to enlarge the dataset. In addition to annotating additional questions, it would also be possible to show annotators questions with low agreement to decide on these cases. This would require a modification of the annotation tool.

4.3 Dataset Filtering

In order to ensure a high-quality dataset, we require two annotations per question by different annotators and filter out questions with low inter-annotator agreement (IAA). To this end, we measure question-level agreement using the Jaccard index over the two (possibly empty) sets of sentences judged as relevant to answering the question by the two different annotators. A formal introduction to the Jaccard index and chance correction is provided in Section 2.3.1.

We determine a threshold for the minimum required agreement for our gold-standard dataset (Section 4.3.1) and present a heuristic to account for hard-to-draw boundaries between actual answers and helpful context (Section 4.3.2). Finally, we give a short overview over the results of the filtering process (Section 4.3.3).

4.3.1 Agreement Threshold

The average IAA over all double annotated questions is 0.34 (chance corrected: 0.31). Due to the definition of the Jaccard index, a disagreement on whether the question is answerable at all, i.e., that exactly one of the two sets of answer sentences is empty, automatically leads to a score of zero. Hence, all questions with such a serious disagreement are removed by applying a threshold > 0.

To assure a high-quality dataset, we filter out questions with a (non-chance-corrected) Jaccard index <0.5. A threshold of 0.5 implies an agreement in more than 50% of all presumably relevant sentences. Manual inspection confirms this threshold as a good balance to ensure high quality and quantity. Compared to Henning et al. (2023) who stipulate the minimum agreement to an F1 score of 0.3 we apply significantly higher standards to our dataset. Since $J(A, B) \leq F1(A, B)$ (proof in Appendix A), we require a more than 65% higher agreement to accept QA pairs into our gold-standard dataset.

4.3.2 Answer Expansion

The relatively low agreement of 0.34 can be partly attributed to the fact that most questions are non-factoid, i.e., answers are not objective single "facts" but instead one or more sentences. This results in sometimes hard-to-draw boundaries what is actually part of the answer and what is just additional context. To account for this difficulty, we modify the annotations in a heuristic way as illustrated in Fig. 4.8. For each sentence marked by just one of the annotators that is adjacent to a sentence marked as relevant by both annotators, we change the annotation of the respective other annotator to "relevant" as well. We do this only if the sentence originally marked by both annotators is no more than three sentences away. We choose the threshold of max. three adjacent sentences as a middle ground between too little and too much additional context backed up by a manual inspection of samples. After modifying the annotations to include adjacent sentences, the average Jaccard index is 0.50, which constitutes an improvement of around 0.16.

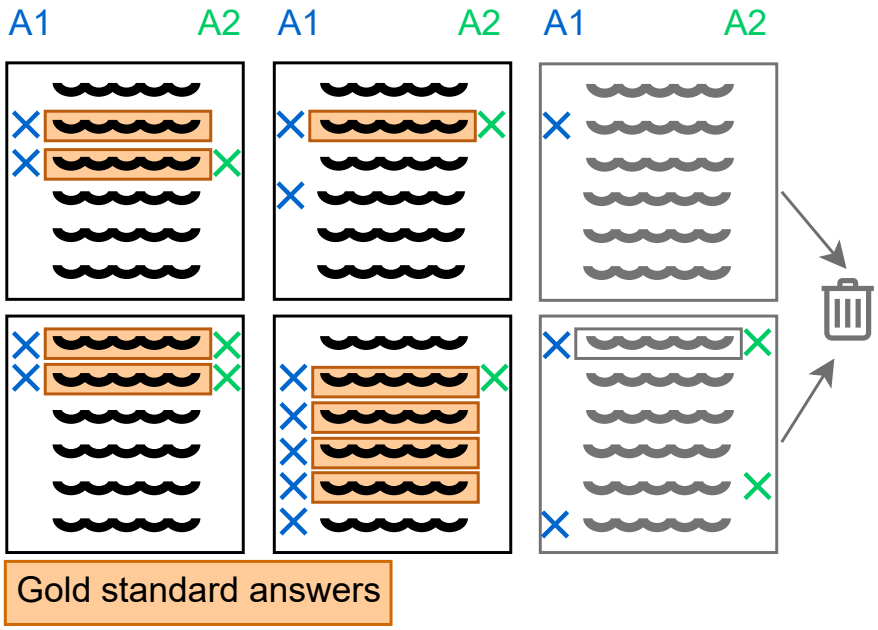

Figure 4.8: Ground-truth answer construction from answer annotations of two human annotators A1 (blue) and A2 (green). The gold-standard contains sentences that A1 and A2 both mark as answers, as well as adjacent sentences marked by only one of them if at most three sentences away from the agreed-upon answer.

	Unfiltered				Filtered			
	#Q.	#Disagr.	#Low Agr.	Jaccard	#Q.	#Answer	#No Answer	Jaccard
w. evidence	760	70 (9%)	250 (33%)	0.32	440 (58%)	436 (99%)	4 (1%)	0.84
w/o. evidence	984	310 (32%)	208 (21%)	0.31	466 (47%)	296 (64%)	170 (36%)	0.88

Table 4.1: Inter-annotator agreement on questions generated with and without evidence on both the unfiltered and filtered dataset. Questions with disagreement regarding its answerability (*#Disagr.*) and those with too little agreement on the answer sentences (*#Low Agr.*) are listed separately. Jaccard index is chance corrected.

4.3.3 Dataset Filtering Results

The applied threshold of 0.5 together with the answer expansion with up to three adjacent answers leaves us with 906 questions (German: 663, English: 243). This amounts to 52% of all double annotated questions. The mean agreement of annotators on the filtered dataset is 0.86 (chance corrected: 0.86). The agreement when leaving out the expansion of including adjacent sentences amounts to 0.61 (chance corrected: 0.59).

As shown in Table 4.1 average agreement is nearly the same between questions generated with evidence and those without (0.31 and 0.32 respectively). However, this similar IAA is deceptive: While out of the 760 questions generated with evidence 58% meet our requirements, this is only true for 47% of the 984 questions created without evidence. For the latter annotators disagree in 32% of all cases whether the question has an answer at all and have to low agreement for a further 21% of questions. Due to the high number of unanswerable questions, which per definition have an IAA of 1, the average agreement is still only slightly lower. On the filtered dataset the reverse is the case with a slightly higher IAA for questions generated without evidence (0.88 compared to 0.84).

As expected, nearly all questions generated with the full document in the prompt meeting our requirements are annotated as answerable (436; 99%). However, even for the questions generated from a short three-word topic most questions (296; 64%) in the gold-standard dataset are deemed to be answerable by our annotators. Even for questions generated with evidence that are expected to have an answer some annotators found no answer (9%) but only rarely both annotators agreed on this (1%). Further research could investigate this for potential bias of annotators to try to find answers even if there are none. In general, annotators seem to struggle with deciding whether a question is answerable.

Additional statistics on our dataset are presented later on in this chapter in Section 4.6. As gold-standard answers we choose the intersection of both annotations, but including adjacent sentences as explained above.

Possible future improvements in the dataset filtering process include filtering out annotators with low overall agreement. Including initial or randomly blended test cases into the regular questions during the annotation process to further distill trustworthy authors would be possible.

4.4 Translations

To increase the size of the dataset and to take the multilingual setting into account, we translate the German questions and documents to English and vice versa using DeepL. [14] While translations for all documents are available in the Integreat-App, they are not usable for our dataset since they are translated on a per-document basis. Translating a full document at once interferes with the mapping from sentence indices to actual sentences due to the asymmetrical nature of language translations. In order to preserve the gold-standard answers represented by the sentence indices, we translate each document sentence-by-sentence. Accordingly, in the German version of the dataset 240 and in the English version 666 of the 906 questions are machine-translated. We retain the information on the original languages for our experiments and for future reference. We have checked random samples of the translations and found them of high quality.

In addition to the translation of German and English QA pairs in both directions, we also translate the dataset to French, Arabic, and Ukrainian. We also use DeepL for this purpose and translate sentence-by-sentence. We plan to use these translations in several multilingual experiments.

4.5 Dataset Split

We split our dataset into train (51%), dev (21%), and test (28%) partitions with similar internal distributions for the original language and the city the document is from. We refer to questions as contiguous if all answer sentences are connected, i.e., with no non-answer sentences in between. In addition to the desirable distributions mentioned above we also aim for evenly distributed unanswerable questions, questions with contiguous answers, and non-contiguous ones among all partitions. As some questions refer to the same document, we make sure that no document occurs in multiple partitions.

To determine a split fitting our criteria we create bins for all possible city-language pairs and assign the matching questions. We then randomly partition each of those six bins into three partitions and experiment with the random seed until we receive a satisfactory result with similar properties.

[14] `https://developers.deepl.com/docs`

Question Word	Count	%	Question Word	Count	%
What	445	49	How	154	17
Where	96	11	Who	74	8
Which	43	5	Yes/No	30	3
When	15	2	Other	49	5

Table 4.2: Distribution of question words in our dataset. German question words are translated.

The partitions are created at document level instead of question level to prevent documents from appearing in multiple partitions. The proposed split is ensuring a close to uniform distribution of several key properties of the dataset such as the agreement of both annotations, the document length or the annotated answer count (see Table 4.3).

4.6 Final Dataset and Corpus Statistics

Our final dataset consists of 906 diverse and high-quality QA pairs. Table 4.2 shows the distribution of used question words among all questions. There seems to be a bias towards "What" with around half of all questions (49%). Other frequently used question words are "How", "Where" and "Who" accounting to another 36% of our dataset.

Out of the 906 QA pairs included in our final dataset, 151 (17%) have non-contiguous answers, 110 (12%) have a single answer sentence and 174 (19%) questions have no answer in the document. The IAA did not differ substantially between German and English annotations in both the unfiltered dataset (German: 0.34, English: 0.32) and the final dataset (German: 0.86, English: 0.86). The same applies for the standard deviation of the IAA, which is almost the same for both languages. However, answers are less frequently non-contiguous for English QA pairs (8% compared to 20%).

Annotators selected notably more answer sentences per question in German (5.39) compared to English (4.24). While English documents consist of fewer sentences and are slightly shorter in general, this difference still holds true from a relative perspective: Only 22% of all sentences and therefore 5 percentage points less than in German are deemed answer sentences in English documents. Multiple questions for the same document are more common in our German dataset part.

In general, however, there are no severe differences between German and English questions in our dataset. The same applies for the corpus statistics between our dataset partitions.

Table 4.3 provides an overview of the corpus statistics of the final version of OMoS-QA.

4.7 Discussion

The presented approach leveraging a combination of automatic QG and manual answer annotation provides good results. We thus show that crowdsourcing can greatly facilitate the dataset creation process in providing high-quality QA pairs with high agreement—even if the individuals are untrained and unfamiliar with the topic.

However, as an initial attempt shows, leveraging crowdsourcing for dataset construction also has limitations. We asked human volunteers to come up with questions and to provide references to the corresponding answer(s) in the Integreat-App. We present this approach, which was discontinued

		train	dev	test	total
German	Questions	338	143	185	666
	No Answer	63 (19%)	30 (21%)	43 (23%)	136 (20%)
	Contiguous Answer	209 (62%)	86 (60%)	104 (56%)	399 (60%)
	Non-Contiguous Answer	66 (20%)	27 (19%)	38 (21%)	131 (20%)
	Documents	205	90	117	412
	Questions/Document	1.65	1.59	1.58	1.62
	Sentences/Document	27.16 ± 20.11	27.96 ± 15.87	26.91 ± 17.88	27.26 ± 18.59
	Chars/Sentence	58.62 ± 15.93	61.74 ± 16.32	61.96 ± 17.25	60.25 ± 16.44
	Chars/Question	57.85 ± 15.68	58.91 ± 17.21	59.61 ± 16.45	58.56 ± 16.23
	Agreement (Jaccard)	0.60 ± 0.33	0.59 ± 0.33	0.60 ± 0.34	0.60 ± 0.33
	with adjacent sentences	0.86 ± 0.19	0.85 ± 0.18	0.86 ± 0.19	0.86 ± 0.19
	Answer Sentences/Question	5.37 ± 6.09	5.57 ± 5.89	5.29 ± 6.84	5.39 ± 6.26
	Answers Sentences/Total Sentences	0.28 ± 0.29	0.25 ± 0.27	0.27 ± 0.28	0.27 ± 0.28
English	Questions	123	50	67	240
	No Answer	18 (15%)	8 (16%)	12 (18%)	38 (16%)
	Contiguous Answer	95 (77%)	38 (76%)	49 (73%)	182 (76%)
	Non-Contiguous Answer	10 (8%)	4 (8%)	6 (9%)	20 (8%)
	Documents	103	43	59	205
	Questions/Document	1.19	1.16	1.14	1.17
	Sentences/Document	23.51 ± 13.30	25.58 ± 16.68	25.49 ± 13.68	24.52 ± 14.14
	Chars/Sentence	65.28 ± 18.22	61.74 ± 12.72	60.48 ± 15.30	63.16 ± 16.45
	Chars/Question	59.46 ± 15.98	56.48 ± 13.22	56.51 ± 14.72	58.01 ± 15.11
	Agreement (Jaccard)	0.58 ± 0.34	0.59 ± 0.32	0.56 ± 0.34	0.58 ± 0.34
	with adjacent sentences	0.86 ± 0.20	0.84 ± 0.19	0.86 ± 0.20	0.86 ± 0.19
	Answer Sentences/Question	4.41 ± 4.98	3.90 ± 3.62	4.19 ± 4.39	4.24 ± 4.55
	Answers Sentences/Total Sentences	0.23 ± 0.23	0.20 ± 0.21	0.22 ± 0.24	0.22 ± 0.23
All	Questions	461	193	252	906
	No Answer	81 (18%)	38 (20%)	55 (22%)	174 (19%)
	Contiguous Answer	304 (66%)	124 (64%)	153 (61%)	581 (64%)
	Non-Contiguous Answer	76 (16%)	31 (16%)	44 (17%)	151 (17%)

Table 4.3: Overview of corpus statistics of the final OMoS-QA dataset. The Jaccard index is chance-corrected.

due to unsatisfactory results, together with possible explanations in Section 4.7.1. Furthermore, we discuss several considerations in regard to the presented dataset construction process in Section 4.7.2 and the annotation tool Section 4.7.3.

4.7.1 Question Crowdsourcing Approach

For the initial crowdsourcing approach we built a form with input fields for question, answer, and reference to the answer document in the Integreat-App. Multi-part answers were taken into account by asking for answer completeness and showing additional input fields for up to four more partial answers if needed. Participants were allowed to complete the form multiple times and all responses were stored anonymously. A screenshot of the form can be found in Appendix E. We targeted civil servants and employees of NGOs working in the integration and migration counselling context as well as volunteers, mostly with a history of supporting refugees and migrants.

Results Over a course of more than two months, the form was only completed 36 times. At the same time, the form was viewed 287 times, which computes to a completion rate of only 13%. While all but one of the responses generally follow the instructions, the approaches and the quality

of the responses are very mixed: In 12 cases (33%) the entered questions consist of only keywords, e.g., "Deutschkurs" or "trans". 8 responses (22%) do not properly include the reference to the source of the answer. Furthermore, out of the cases that do include a reference, the entered answer text is not correctly reproduced in 13 cases (36%). 9 times (25%) a second answer, 7 times (19%) a third answer, and 4 times (11%) a fourth answer is included in the response. A few examples of responses are shown in Appendix E.

All in all, this approach produces only very few QA instances and even fewer of sufficient quality. It is therefore not pursued further and the responses are not used.

Problems and Explanations Based on the completion rate of only 13% as well as personal feedback, the task of manually collecting QA pairs seems to be too challenging and time-consuming for untrained volunteers. This can be attributed to the various different steps involved as well as to too few constraints and not enough assistance being put in place to aid in completing the form. The task involves looking up different pages of the Integreat-App, coming up with a question, copying the URL of the page to the form, narrowing down the answer, and entering it in the corresponding input field. The last step is made more challenging by the use of HTML instead of plain text in the content of the Integreat-App. Deciding whether the answer is complete and providing additional answers and references in the adverse case posed a further source of complexity.

Form various inquiries about the tasks and the low quality of the responses, we conclude that the instructions were too vague and unspecific to allow for good results. At the same time, the input fields were not constrained enough to prevent human error, which complicates postprocessing and interpretation of the responses. Additional information or training of the volunteers could have mitigated this but was not possible in the scope of this work. Some volunteers found the task of completing the form redundant due to the impression that all the questions and answers "can already be found in the app".

Finally, this approach only yields data points with evidence, while for the application also instances without evidence are necessary. As discussed in Section 3.2.2, phrasing diverse high-quality questions is difficult for humans knowing the complete text, especially questions without evidence.

4.7.2 Answer Expansion and Thresholds

The answer expansion approach (Section 4.3.2), which is implemented to partly mitigate some difficulties of non-factoid QA, e.g., defining the boundaries of answers, could be evaluated and refined further. At the moment, we always only add the answer sentences annotators disagree on, given that they agree on an adjacent sentence. We therefore increase the amount of answer sentences compared to the actual agreement between annotators. This might have implications on model performance by leading to a higher precision and lower recall for extracting answer sentences. By expanding our answers before the dataset filtering step, we also include questions in the final dataset that would otherwise fail the threshold for IAA.

In general, the used IAA threshold can be adjusted to either increase the quantity or the quality of the dataset. We chose a Jaccard index of 0.5 as threshold as a good balance between quality and quantity (Section 4.3). However, it is possible to lower this value to increase the size of the final dataset or raise it to ensure an even higher inter-annotator agreement and therefore quality.

4.7.3 Annotation Tool Considerations

We considered several other changes in regard to the human annotations and the custom annotation tool (Section 4.2.2). We rejected the idea of preselecting sentences based on QA results from a model or, if existing, previous human annotation. This could ease the annotation process and make it more efficient for annotators, as there is already a hint on answer sentences, and usually fewer checkboxes have to be selected. However, this could introduce potentially harmful biases, such as annotators only confirming the preselection and not actually considering all sentences. We therefore conclude, that the possible savings in time do not outweigh the introduction of biases. However, the annotation tool can be adjusted to enable this feature.

Originally, we planned to allow users to select the source of the questions to receive, i.e., the city of the Integreat-App knowledge base the document is from. Since we decided to employ a crowdsourcing approach with volunteers, which usually live in different cities than the ones our documents originate from, we removed this feature. For human experts from the authorities or NGO of a specific city, this feature could be enabled to allow for a more fine-grained question selection.

4.8 Conclusion

As a conclusion, dataset construction can be supported and simplified by the use of both NLP and human crowdsourcing approaches. By splitting the dataset creation into separate modular steps, we can choose and combine the best solutions for the different tasks. Attempting to create both questions and to provide answers in the same step produces suboptimal results.

We show that crowdsourcing can provide high-quality annotations, even with untrained volunteers. However, in order for crowdsourcing to produce good results the task at hand has to be simple, constrained, and efficient, such that there is little room for wrong interpretation or human error. Easy-to-use tools and single-step tasks allow for a higher participation with better results. In contrast, more open tasks like question generation require specific training or experienced crowd workers. Alternatively, we show that automatic QG leveraging LLMs can provide good results as a compromise.

The introduced dataset construction attempt is modular, scalable, and requires minimal human effort while still assuring high-quality results. The dataset is extractive and includes the sentence indices of answer sentences. Our final dataset consists of 906 high-quality QA pairs on Integreat-App documents in German and English.

5. Modeling

We conduct both sentence-level experiments to extract answer sentences and question-level experiments to detect whether a question is answerable at all. For each of those experiments, we consider two different setups: Binary classification using a finetuned DeBERTa and a generative approach leveraging various LLMs. The main focus lies on open-weight LLMs by MistralAI and Meta. Additionally, we include the closed-source and closed-weight GPT-3.5-Turbo by OpenAI for comparison.

In this chapter, we first summarize the used training data and the common prerequisites of our dataset (Section 5.1). We then describe the classification setup for both sentence-level answer extraction (Section 5.2.1) and question-level unanswerability detection (Section 5.2.2). Finally, we present the generative setups on a sentence-level (Section 5.3.1)

5.1 Dataset and Prerequisites

We train and evaluate our models on our OMoS-QA dataset. Most experiments are carried out on the German and English versions of the dataset consisting of 906 QA pairs described in Section 4.6. For some multilingual experiments we additionally use machine translated versions of OMoS-QA in Arabic, French and Ukrainian. We use the training partition to finetune DeBERTa and for sampling QA pairs for the LLM 5-shot setting. The development set is used to evaluate both the finetuning progress of the binary classifiers and the prompts for the LLMs. Finally, we report results of our experiments on the test partition.

The OMoS-QA dataset provides sentence-level annotations for whether a sentence provides information that is relevant to answering a question Hence, answers to the question are given as list of sentence indices of the document. Unanswerable questions are indicated by an empty sentence indices list. Correspondingly, our experiments are also extractive on a sentence level.

5.2 Classification Setup

We consider two different classification setups for our experiments: First, we try to extract answer sentences by binary sentence classification (Section 5.2.1). Second, we train a binary classifier to predict whether a question is answerable at all, i.e., to decide if the document contains any answer to the question or not (Section 5.2.2). We describe the used setups including contexts, special tokens, instructions and hyperparameters.

5.2.1 Answer Extraction by Sentence Classification

In this setup, we extract answer sentences using binary sentence classification on each individual sentence of our document. We pass the question, the current sentence and additional context to the model. To decide on the optimal context window size, we conduct finetuning experiments (Section 6.2). The setup is pictured in Fig. 5.1.

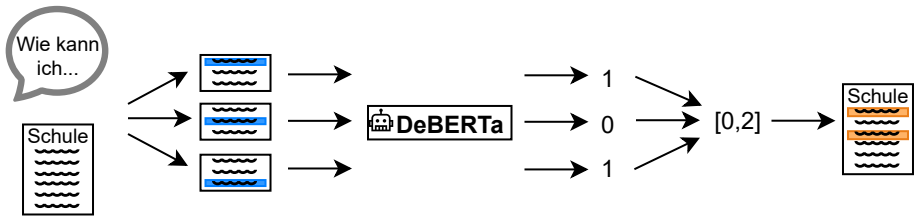

Figure 5.1: Setup for answer extraction by sentence classification. The classifier considers each sentence individually. Model output is *1*, if the answer sentence is considered an answer, and *0* otherwise. The currently considered sentence is highlighted blue.

```
[CLS] what is the lowest language level?
[SEP] a1: beginner they can understand and use simple words and sentences.
[SEN] you can introduce yourself and others. [SEN]
for example: my name is maria. i am 30 years old. [SEP]
```

Figure 5.2: Instruction format for answer sentence extraction with finetuned DeBERTa. We add a new special token `[SEN]` to our vocabulary to mark the currently considered sentence. The currently considered sentence for classification is highlighted blue.

Model. We attach a binary classification head on the pretrained DeBERTa (Section 2.1.2.2) model using the Hugging Face Transformers framework.[1] The binary head is a linear layer with 1024 input and 2 output features on top of a pooling layer. We separately finetune on OMoS-QA in all languages and the retranslated dataset versions. The model and the finetuning is evaluated according to its F1-score.

Data. For the binary classification of sentences we consider one sentence at a time. We therefore split our OMoS-QA into question-sentence pairs and consider every sentence s_{i_j} for a document d_i separately. Our train, development and test partitions for answer sentence classification therefore consist of significantly more instances than the original dataset with 11,812, 5,241 and 6,391 instances respectively. Surrounding sentences are provided as additional context in the instruction.

Instruction Format and Special Tokens. We use the recommended classification instruction format for models based on BERT with the default special tokens for classification `[CLS]` and separation `[SEP]` as shown in Fig. 5.2. While the input is prefixed with `[CLS]`, the `[SEP]` token separates the question from the sentence (and additional context) and concludes the instruction. The predicted classification of the sentence is computed from the final embedding of the classification (`[CLS]`) special token. In addition, we add a new special token `[SEN]` to our vocabulary to separate the additional context and highlight the actual current sentence. An example instruction is shown in Fig. 5.2.

We also tested finetuning the model without the new `[SEN]` special token. Performance (F1-score) on the development dataset was around 3 percentage points lower than with this new token (English, context window size of 3).

Hyperparameters. The hyperparameters used for finetuning are listed in Table 5.1.

[1] `https://huggingface.co/docs/transformers`

	Sentence Classification	Question Classification
Batch size	8	8
Learning rate	$2 * 10^{-6}$	$2 * 10^{-6}$
Weight decay	0.1	0.1
Warmup steps	50	50
Evaluation steps	50	10
Max. epochs	3	10
Early stopping	10	10

Table 5.1: The used hyperparameters for finetuning DeBERTa for answer extraction using binary sentence classification and question answerability classification.

```
[CLS] {question} [SEP] {document} [SEP]
```

Figure 5.3: Embedding for question answerability classification with finetuned DeBERTa.

Observations. A higher learning rate ($2 * 10^{-5}$) causes the model to overshoot without making any progress in learning. Precision, recall and F1-score are staying nearly consistently low while the loss is continuously increasing as the model is making both correct and incorrect predictions with increasing confidence.

5.2.2 Question Answerability Classification

In addition to detecting specific answer sentences, we conduct experiments to detect unanswerable questions, i.e., questions that are not answerable given the document. We also use a finetuned binary classifier to decide whether a question is answerable or not. We pass question-document pairs directly to the model without the need for additional special tokens or context windows (Fig. 5.3).

In comparison to answer extraction by binary sentence classification, most of the setup is analogous. Since we have less data instances in comparison, we increase the number of maximum number of epochs to 10 and decrease the steps between evaluation to 10. All used hyperparameters are shown in Table 5.1.

Observations. Because of memory issues with too long documents we just truncate the instructions to a maximum length of 1024. Long documents are therefore not completely represented. In further experiments, a sliding window can be employed.

Due to the unbalanced distribution of answerable and unanswerable instances and the use of precision, recall and F1-score to measure performance, it is necessary to evaluate model performance on the minority class, i.e., unanswerable instances. Otherwise, the model starts and idles with a recall of 1 and a precision of 0.81 by classifying all questions as answerable.

5.3 Generative Setup

In addition to a classification approach, we also want to evaluate the performance of LLMs on our QA task. As mentioned above, LLMs are generative and create new text. While this is desirable for a lot of use cases, it also comes with several problems, such as possible hallucinations and

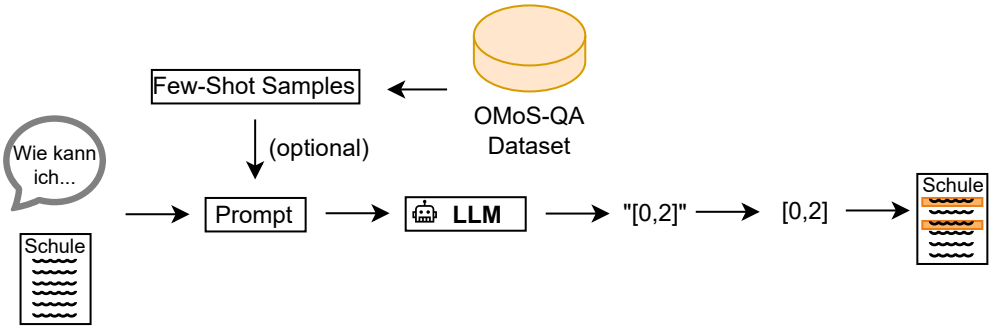

Figure 5.4: Setup for answer sentence extraction by index generation. The model is prompted to generate a list of indices of answer sentences in the document. The generated text is postprocessed and sentence indices are extracted. In the few-shot setting, five samples from the train partition of our OMoS-QA dataset are included in the prompt.

discrimination as well as missing reproducibility and trustworthiness. Due to the context of OMoS, we try to minimize that risk and attempt a "non-generative" approach. We use generative LLMs in an extractive manner by prompting the models to output the indices of sentences that are part of an answer to the question. If a question is not answerable, the model is instructed to return an empty list. This approach is similar to the one followed by Henning et al. (2023) and is described in Section 5.3.1.

Furthermore, we conduct separate experiments to detect unanswerable questions. In addition to the classification approach with DeBERTa described earlier, we also try a generative setup leveraging LLMs. The corresponding setup and the used prompts are explained in Section 5.3.2.

5.3.1 Answer Sentence Extraction by Index Generation

Model Setup. We use the *text generation pipeline* from Hugging Face Transformers for inference.[2] We tweak the *temperature* hyperparameter to 0.75 to decrease the variance of model outputs allowing for easier postprocessing. All models used are introduced in Section 2.1. We show the setup for answer sentence extraction by index generation in Fig. 5.4.

Prompt. We mostly follow the prompt templates proposed by Henning et al. (2023) for both the zero-shot and 5-shot settings. It instructs the models to output a list containing the sentence indices of the answer sentences, e.g., `[1,2,3]`. If the question is not answerable, the model is ordered to output an empty list (`[]`). While Mistral-7B and Mixtral-8x7B only support a simple classical instruction format including only prompt and model answers, Llama-3-8B and Llama-3-70B allow for a more sophisticated chat-like format with *system*, *user*, and *assistant* messages. Example instructions are shown in Fig. 5.5 for Mistral-7B and Mixtral-8x7B, and in Fig. 5.6 for Llama-3-8B and Llama-3-70B. The prompts are shortened in both examples with "...".

In the 5-shot setting, we additionally include five manually selected and chunked examples with question, document chunk and the expected model output. The examples consist of three answerable and two unanswerable questions from the train partition, which is otherwise unused in our LLM experiments. The sampled QA pairs are selected to represent diverse answers, including single sentence, contiguous, and non-contiguous answers. For each sample we include the corresponding

[2] `https://huggingface.co/docs/transformers/main_classes/pipelines`

```
<s>[INST] Given the question and document below, select the sentences from the
document that answer the question.
It may also be the case that none of the sentences answers the question.
In the document, each sentence is marked with an ID.
Output the IDs of the relevant sentences as a list, e.g., "[1,2,3]", and output
"[]" if no sentence is relevant.
Output only these lists.

Question: Where can I request a certified translation?

Document: [0] Professional interpreting and translation by specialists
[1] You don't speak German very well yet?
[2] Then an interpreter can help you in appointments.
[3] For example, at the immigration office or the education authority.
...
[/INST]
```

Figure 5.5: Zero-shot prompt for Mistral-7B and Mixtral-8x7B for answer sentence extraction.

prompt analogous to the zero-shot prompt and the expected model response with the answer sentence indices. While we use the same examples for every model and question, we use the respective translation for German and English QA. More information on the 5-shot prompt and the used samples can be found in Appendix C.

Prompt Engineering. We have experimented with different prompts on the development set aiming for high-quality predictions. Apart from the precision and recall of the predictions we also desire uniform outputs to allow for easy postprocessing. In our first iteration of the prompt we have tried to instruct the model to use the following less formal pattern among others:

```
## Answer: {answer} ## Sentence numbers: {answer sentence numbers}
```

The complete prompt can be found in Appendix C.2. Compared to our final prompt, model outputs are notably less uniform and therefore harder to postprocess.

We have tested different phrasings of the prompts as well as different ways of passing the document. Additionally, we have tried both enumeration of answer sentences with just numbers and numbers in brackets (`[3]`). The phrasing we used in our final prompt together with wrapping sentence indices in brackets produce the best results.

5.3.2 Question Answerability by Text Generation

Model Setup. The model setup is analogous to Section 5.3.1.

Prompt. The prompt is set up similar to Section 5.3.1 including question and the complete document and converted to the respective instruction formats of the models. However, instead of instructing the model to generate indices of answer sentences, we ask the model to respond with either `[YES]` or `[NO]`, depending on whether the question is answerable or not. An example of the prompt for the Llama-3 models can be found in Fig. 5.7.

```
<|im_start|>system
Your task is to select sentences from a document that answer a given question.
<|im_end|>
<|im_start|>user
Given the question and document below, select the sentences from the document
that answer the question. It may also be the case that none of the sentences
answers the question. In the document, each sentence is marked with an ID. Output
the IDs of the relevant sentences as a list, e.g., "[1,2,3]", and output "[]" if
no sentence is relevant. Output only these lists.

Question: Where can I request a certified translation?

Document: [0] Professional interpreting and translation by specialists
[1] You don't speak German very well yet?
[2] Then an interpreter can help you in appointments.
[3] For example, at the immigration office or the education authority.
...
<|im_end|>
<|im_start|>assistant
```

Figure 5.6: Zero-shot prompt for Llama-3-8B and Llama-3-70B for answer sentence extraction.

```
<|im_start|>system
Your task is to decide whether the document answers the question.
<|im_end|>
<|im_start|>user
Does the document below contain an answer to the question? If the document
contains an answer, output "[YES]". If the document does not contain an answer,
output "[NO]". Do NOT output any additional text.

Question: Where can I request a certified translation?

Document: [0] Professional interpreting and translation by specialists
[1] You don't speak German very well yet?
[2] Then an interpreter can help you in appointments.
[3] For example, at the immigration office or the education authority.
...
<|im_end|>
<|im_start|>assistant
```

Figure 5.7: Zero-shot prompt for Llama-3-70B for explicit question unanswerability detection.

6. Experiments

In this section, we describe our experiments. We evaluate several off-the-shelf LLMs and a pretrained classifier on OMoS-QA in various settings. We focus on open-weight models from MistralAI and Meta and provide results from GPT-3.5-Turbo and finetuned DeBERTa for comparison. Additionally, we evaluate the models against human agreement estimated from the agreement in the manual answer annotations from our dataset construction. Apart from classical question answering we conduct multilingual experiments to determine the multilingual capabilities of the models and compare those to leveraging machine translation before prompting.

We first describe the evaluation methods and human agreement measures derived from IAA in Section 6.1. We then present our conducted experiments and their results. An experiment to determine the optimal context window size for answer extraction using sentence classification with DeBERTa is described in Section 6.2. We then compare different LLMs in zero-shot and 5-shot settings with DeBERTa and human agreement for German and English OMoS-QA (Section 6.3). We also evaluate models on OMoS-QA translated to additional languages, which are relevant in the migration context, and compare those results with retranslating to German (Section 6.4). In addition, we conduct pilot experiments with cross-language QA pairs (Section 6.5) and explicit unanswerability detection (Section 6.6). The results of our experiments are finally summarized in Section 6.7.

6.1 Evaluation and Human Agreement

In this chapter, we describe the evaluation of our answer sentence extraction experiments (Section 6.1.1). We then derive an estimation of human agreement from the IAA of our human annotations, against which the models are evaluated (Section 6.1.2).

6.1.1 Evaluation

For all of our experiments, we evaluate model performance using precision, recall, and F1-score. While the Jaccard index only compares the agreement of two sets symmetrically, the F1-score is derived from precision and recall and therefore takes the relation of predictions and ground-truth answer into account. Additionally, the F1-score is commonly used to measure model performance in related work (Henning et al., 2023; Prasad et al., 2023; Wang et al., 2019b). In the following, we denote the F1-score as F.

We conduct answer sentence extraction experiments using LLMs and binary classifiers. The extracted answer sentences of a model m for a set of question-document pairs S, usually the test partition, are represented by a list of sentence indices Y_{m_S}. The ground-truth answers for S are denoted as X_S and $|S|$ is the number of instances in S. We calculate agreement on a sentence level, i.e., we measure the agreement between model predictions Y_{m_S} and ground-truth answers X_S. Additionally, we evaluate question-level unanswerability agreement by inferring unanswerability from an empty list of extracted sentence indices.

Sentence-level Evaluation. Our evaluation consists of computing precision and recall of the extracted sentence indices against the ground-truth answers of OMoS-QA separately for every ques-

tion. We then macro-average these scores over all questions of the considered partition S. Finally, we compute the F1-score from the macro-averaged precision and recall. Sentence-level macro-averaged model precision P_m and recall R_m can be calculated as follows with Eq. (2.5) and Eq. (2.6):

$$P_m = \frac{\sum_{d \in S} P(X_d, Y_{m_d})}{|S|} \tag{6.1}$$

$$R_m = \frac{\sum_{d \in S} R(X_d, Y_{m_d})}{|S|} \tag{6.2}$$

F_m is then calculated from P_m and R_m with Eq. (2.7).

Question-level Evaluation. While most of our experiments produce a list of answer sentence indices, we additionally evaluate these sentence-level results on a question level. We infer question (un)answerability trivially from the list of answer sentence indices: If the list of answer sentence indices is empty, the question is unanswerable given the document. Otherwise, the question is answerable. In other words, if any sentence is marked as an answer, the question is answerable. We therefore define the sets of retrieved unanswerable questions $\hat{S}_{retrieved}$ as questions with no extracted answer sentences and actually unanswerable questions $\hat{S}_{relevant}$ as questions with no ground-truth answers:

$$\hat{S}_{retrieved} = \{d \in S \mid Y_{m_d} = \emptyset\} \tag{6.3}$$

$$\hat{S}_{relevant} = \{d \in S \mid X_d = \emptyset\} \tag{6.4}$$

We then calculate question-level precision $\hat{P}_m$, recall $\hat{R}_m$, and F1-score $\hat{F}_m$ with Eq. (2.5), Eq. (2.6), and Eq. (2.7).

6.1.2 Human Agreement

In order to put the results of the models in the different setups into perspective, we calculate approximate human agreement for comparison. We derive the human agreement from the inter-annotator agreement of our annotators in our dataset construction process (Section 4.6). In contrast to the results of our experiments, there are no ground-truth answers with which the annotations could be compared, since both annotations are equally likely to be correct. Therefore, precision and recall are symmetric and interchangeable. However, since $Precision(A, B) = Recall(B, A)$ holds true and the F1-score is therefore symmetric, i.e., $F1(A, B) = F1(B, A)$, we can still provide a F1-score: For each question, the data labeled by the various annotators is assigned to one of two sets randomly, and then one set is treated as the gold standard and one as human predictions.

We provide this human agreement in two different variants: Agreement on the unfiltered dataset accounts to 57.8% and on the test partition, we measure a human agreement of 76.3%. Agreement is calculated without expanding answers as explained in Section 4.3.2. As the German and English version of the dataset consist of the same (potentially translated) questions and documents, the score is the same. Due to the random assignment of annotations to either of the two sets, the F1-score is only an approximation of human agreement.

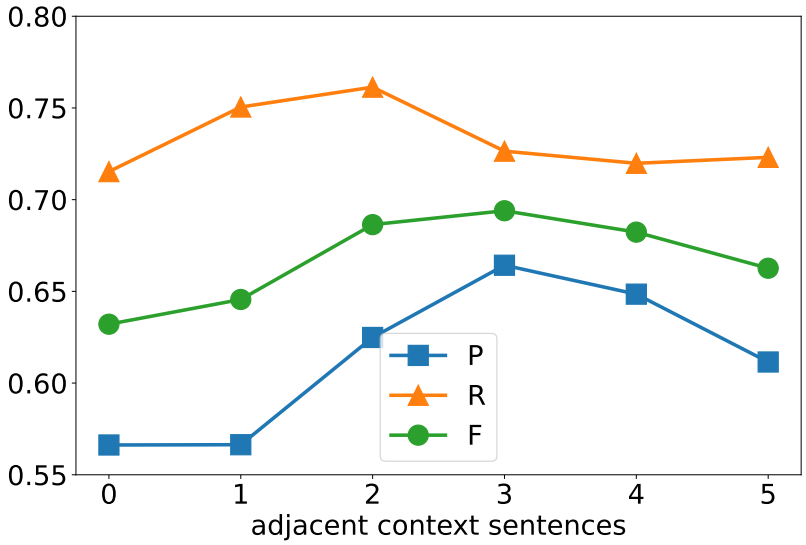

Figure 6.1: Precision (P), recall (R) and F1-score (F) of finetuning DeBERTa with different context window sizes for sentence classification for answer extraction on the English development partition.

Since the identification of unanswerable questions is a binary task on a question level and only QA pairs with agreement on their answerability are included in the final dataset, calculating a human agreement for this task on the filtered dataset result in a score of 100%. On the unfiltered dataset, the human agreement on question-level answerability accounts to 47.8%. Similarly to human agreement on the sentence level, this is not directly comparable to model performance.

6.2 Context Window for Sentence Classification

We finetune DeBERTa with various context window sizes around the current sentence s_{i_j} of a document d_i. We try context window sizes w between 0 and 5 indicating the maximum amount of preceding and succeeding sentences respectively. If less than w sentences are preceding or succeeding the sentence s_{i_j}, i.e., $j < w$ or $j > |d_i| - w$, fewer sentences are given as context without any replacement. The model setup and the instruction format are described in Section 5.2.1 and Fig. 5.2.

Fig. 6.1 shows the finetuning results on the development partition with aforementioned context window sizes. While the recall already starts to decrease for context window sizes > 2, maximum precision is reached with a context window size of 3. The F1-score combining precision and recall also reaches its maximum with a content window size of 3. As precision is of particular importance for us and the F1-score also peaks with that context window size, we decide on a size of 3 adjacent sentences as context for all following classification experiments.

6.3 Answer Sentence Extraction

Our first experiment is the extraction of answer sentences in our OMoS-QA dataset. For LLMs we use text generation of answer sentence indices and prompt our models with the question and the document with enumerated sentences. The models are instructed to output a list of answer

		German				English			
Model	Setting	Pattern	Add. Text	Inv.	Unansw.	Pattern	Add. Text	Inv.	Unansw.
Mixtral-8x7B	0-shot	83.4	52.6	14.2	12.6	84.0	56.2	14.4	9.2
	5-shot	97.3	5.2	0.7	21.1	97.1	5.2	0.4	21.6
Mistral-7B	0-shot	96.9	44.3	1.8	4.5	96.4	34.6	2.2	7.0
	5-shot	42.7	83.1	9.2	13.0	50.1	78.4	6.5	13.7
Llama-3-8B	0-shot	87.0	97.8	13.0	18.9	93.3	96.9	6.3	22.7
	5-shot	97.3	96.9	0.0	31.5	97.3	95.1	0.0	31.9
Llama-3-70B	0-shot	100.0	0.0	0.0	20.4	99.3	0.7	0.0	19.3
	5-shot	98.2	1.8	0.0	23.1	98.9	1.1	0.0	24.0
GPT-3.5-Turbo	0-shot	100.0	20.2	2.0	24.2	99.6	23.4	2.0	25.0
	5-shot	100.0	0.4	0.0	21.8	100.0	0.0	0.0	24.2

Table 6.1: Postprocessing results (in %) of LLM responses for answer sentence extraction in zero-shot and 5-shot settings on development and test partitions combined (455 QA pairs). The columns indicate the amount of results in % that follow the bracket pattern (*Pattern*), include additional text (*Add. Text*), are not parsable (*Inv.*), and actually predict questions to be unanswerable (*Unansw.*).

sentence indices. If no evidence is present in the text, the model is instructed to output an empty list. For DeBERTa the model classifies separately for each sentence whether it is (part of) an answer. More details on the setup can be found in Section 5.2 for the classifier and in Section 5.3 for the LLMs.

We first discuss the postprocessing of LLM outputs in (Section 6.3.1). We then present the sentence-level answer results (Section 6.3.2). Subsequently, we infer question-level unanswerability detection performance from the sentence-level results (Section 6.3.3). Finally, using Llama-3-70B, which performs best in most settings, as an example, we compare the performance of the models based on the number of ground-truth answer sentences (Section 6.3.4).

6.3.1 LLM Postprocessing

Since LLMs work in a generative manner, postprocessing is needed to transform the generated output into answer sentence indices for text extraction. We expect a list of sentence indices as responses from the models. Since LLMs often produce additional text, which can pre- and succeed the expected output, we only consider text between the first occurrence of an opening bracket (`[`) and the first occurrence of a closing bracket (`]`). All other text in the response is ignored. In case there are more than one opening or closing bracket, only the first one is taken into account. If the response does not match the pattern at all, i.e., no brackets are present, we use the first line as a fallback and try to interpret the response nevertheless.

We split the extracted text on commas (`,`) and strip surrounding whitespaces and double quotes (`"`). We keep parts consisting only of digits, whitespaces, and hyphens (`-`) while discarding everything else using regular expressions. Sentence indices separated by hyphens are complemented with intermediate sentence indices, e.g., `2-5` is expanded to `2,3,4,5`. The complete code for extracting the predicted answer sentence indices can be found in Appendix F.

Invalid Responses. We treat responses that fail the described postprocessing as prediction that the question is unanswerable given the document. This can possibly lead to false positive detection of unanswerable question. Since we specifically focus on trustworthiness answers, we err on the side of caution and accept this drawback.

Postprocessing Results. Table 6.1 shows the results of the postprocessing for the different LLMs. In general, the LLMs follow the requested pattern of a list of sentence indices in brackets in the majority of cases in the zero-shot setting. Llama-3-8B in German and especially Mixtral-8x7B in German and English are slightly negative outliers with less than 90% of responses adhering to the pattern. Correspondingly, these models have a notable amount of invalid responses (Mixtral-8x7B: 14.2% (de) and 14.4% (en); Llama-3-8B: 13.0% (de) and 6.3% (en)), which are treated as unanswerable. Both Llama-3-70B and GPT-3.5-Turbo produce responses that (nearly) always follow our instructions. However, the latter includes additional text in more than 20% of questions. Nevertheless, this is vastly better than Llama-3-8B close to always ($> 96\%$) and Mixtral-8x7B ($> 52\%$) and Mistral-7B ($> 34\%$) often producing additional text.

In the 5-shot setting, the quality of the responses of all models except Mistral-7B and Llama-3-70B is significantly improved. While for Llama-3-70B there is just a slight decrease in quality compared to near perfect responses in the zero-shot setting, the quality of output from Mistral-7B is much worse. The model adheres to the pattern in less than half of the cases, while additional text is included twice as often. The 5-shot setting reduces invalid output to less than one percent for all models but Mistral-7B, which even sees improved invalidity (de: 9.2%, en: 6.5%).

While providing samples in the prompt appears to be beneficial for instruction following for most models, Mistral-7B seems to be overwhelmed leading to degrading performance. Notably, Llama-3-8B produces additional text for most questions in both the zero-shot and 5-shot setting while still mostly following the pattern and rarely producing invalid responses. While other models mostly include text about the answerability, the question, or the answer sentences, if any, Llama-3-8B mostly continues an imaginary conversation between the user and the assistant.

6.3.2 Sentence-Level Results

We present our sentence-level results for answer extraction in the left half of Table 6.2. All LLMs show good precision (70–88%), with the highest numbers achieved by the Llama-3-70B in both settings, Mistral-7B in the 5-shot, and GPT-3.5-Turbo in the zero-shot setting. Recall is much lower in general, with a wider range across models, reaching as low as 19.5% (Mistral-7B 5-shot German) and as high as 51.7% (Mixtral-8x7B 5-shot German). DeBERTa, the only finetuned classifier among our models, has a notably lower precision (de: 60.3%, en: 63.7%) paired with a significantly higher recall (de: 65.3%, en: 71.0%). This binary classification with DeBERTa apparently performs better for English QA pairs compared to German ones (P: +3.4%, R: +5.7%, F: +4.4%). As DeBERTa was only pretrained on English training data, this is to be expected. While this performance difference gap is also observable in our smaller LLMs (Mistral-7B, Llama-3-8B), Llama-3-70B and especially Mixtral-8x7B even excel on the German QA pairs by an up to 2.5% higher F1-score. It is also worth mentioning the clearly poorer recall of Mistral-7B in the 5-shot setting with a 27.5% and 18.0% gap in German and English respectively, which fits the worsened general responses of the model in the 5-shot setting described in Section 6.3.1.

In total, the binary classification with DeBERTa seems to produce the most balanced results with the best F1-score in both German (62.7%) and particularly in English (67.1%). However, this comes at the cost of a lower precision compared to the LLMs. This behavior (selecting more, only

		Sentence-level Answers						Question-level Unanswerability					
		German			English			German			English		
Model	Setting	P	R	F	P	R	F	P	R	F	P	R	F
Mixtral-8x7B	0-shot	74.5	47.1	57.7	73.4	44.2	55.2	68.9	56.4	62.0	65.8	45.5	53.8
	5-shot	79.0	51.7	**62.5**	77.9	50.5	61.3	67.8	72.7	70.2	65.6	76.4	70.6
Mistral-7B	0-shot	69.7	47.8	56.7	74.1	47.5	57.9	**80.0**	14.5	24.6	70.0	25.5	37.3
	5-shot	**87.6**	20.3	32.9	84.3	29.5	43.7	29.2	**89.1**	43.9	30.3	72.7	42.8
Llama-3-8B	0-shot	74.9	30.0	42.9	78.2	34.8	48.1	71.1	49.1	58.1	54.7	52.7	53.7
	5-shot	81.9	42.2	55.7	82.1	44.2	57.4	54.7	85.5	66.7	53.6	**81.8**	64.7
Llama-3-70B	0-shot	85.5	46.6	60.3	84.8	46.7	60.2	69.8	67.3	68.5	74.5	63.6	68.6
	5-shot	86.7	48.2	62.0	84.9	48.4	61.6	68.3	78.2	**72.9**	64.5	72.7	68.4
GPT-3.5-Turbo	0-shot	85.3	31.6	46.1	**87.3**	31.2	45.9	50.8	60.0	55.0	54.4	67.3	60.2
	5-shot	81.8	45.1	58.1	83.8	43.9	57.6	70.9	70.9	70.9	67.2	74.5	**70.7**
DeBERTa	–	62.6	**62.4**	**62.5**	65.7	**64.2**	**64.9**	56.2	65.5	60.5	59.4	69.1	63.9
*Human Agreement**		–	–	*57.8*	–	–	*57.8*	–	–	*47.8*	–	–	*47.8*
test partition only		–	–	*76.3*	–	–	*76.3*	–	–	*100.0*	–	–	*100.0*

Table 6.2: Test set performance (in %) of zero-shot and 5-shot LLMs and finetuned DeBERTa on sentence-level answer extraction (left) and detection of unanswerable questions (right). The best result in each column is **bolded**. **Human Agreement* is not directly comparable. We compute human agreement from IAA and state numbers on the unfiltered dataset and the test partition. Since only questions with agreement on its answerability are included in the filtered dataset, human agreement is 100.0% for question-level unanswerability.

potentially fitting sentences as opposed to fewer but clearly relevant sentences) might contradict our goals of providing trustworthy results. Future research could attempt to improve finetuning with a focus on increased precision. Both Mixtral-8x7B and Llama-3-70B in the German 5-shot setting show only a slightly lower F1-score with a considerably higher precision, seemingly fitting our goals better. Due to better performance of DeBERTa on English QA, this gap widens to an at least 3.3% better F1-score of DeBERTa compared to LLMs.

The last two rows of the table present approximations of the human agreement, measured as the inter-annotator agreement in our dataset in terms of F1, closely described in Section 6.1.2. Three of our models (Mixtral-8x7B 5-shot, Llama-3-70B in both the zero-shot and 5-shot setting, and DeBERTa) outperform the lower of the two human agreement scores in both German and English. However, the human agreement inferred from the IAA on the unfiltered dataset has two limiting factors: First, our annotators are all untrained volunteers and not human experts. Second, the agreement is calculated on a superset of the test partition, on which models are evaluated, and also include questions with low agreement, i.e., questions that are most likely more subjective and challenging to answer. The human agreement on the filtered dataset, on the other hand, is out of reach for all models with a gap of at least 10%.

6.3.3 Question-Level Unanswerability

Apart from measuring performance of our models on a sentence level, we also infer question-level unanswerability. We measure this identification of unanswerable questions and report the results separately in the right half of Table 6.2. Here, the precision indicates how many of the questions, to

which a model responds with an empty list, actually have no answer in the provided text, according to our annotators. Recall reflects how many of the unanswerable questions are correctly identified by the model as such.

In the zero-shot setting for identifying unanswerable questions, precision is higher than recall for all models except GPT-3.5-Turbo with 9.2% and 12.9% higher recall. Apart from Mistral-7B with F1-scores of less than 25% (German) and 38% (English) due to poor recall (de: 14.5%, en: 25.5%), all models achieve F1-scores higher than 53%. The best and most balanced performance is displayed by Llama-3-70B with scores above 68%. Performance differences between German and English are mixed: While Mixtral-8x7B and Llama-3-8B perform better on the German OMoS-QA, all other models exhibit opposite numbers.

In the 5-shot setting, the picture is different: Recall at least equals and mostly exceeds precision. The biggest gap is again displayed by Mistral-7B (German: 59.9%, English: 42.4%). Mistral-7B's very poor precision and high recall in the 5-shot setting is in line with the observations from the postprocessing with lots of invalid responses treated as unanswerability prediction (compare Section 6.3.1). On the German dataset, Llama-3-70B achieved the best F1-score with 72.9% while suffering a 10% precision drop on the English version. Hence, GPT-3.5-Turbo excels on the English version (70.7%). All models except Mixtral-8x7B perform better on the German version of the dataset.

DeBERTa displays a higher recall than precision on both German and English OMoS-QA, with a gap of close to 10% in both cases and slightly better English numbers. It achieves a higher F1-score than most zero-shot models, but lags behind in the 5-shot setting.

All models except Mistral-7B surpass the human agreement of 47.8% on the unfiltered dataset in both the zero-shot and 5-shot setting. Since this human agreement is calculated on the unfiltered dataset while model performance is evaluated only on the test partition, on whose answerability human annotators fully agreed, the results are not directly comparable. The low human agreement on question (un)answerability indicates that deciding whether a question has an answer sentence at all is already challenging. Table 4.1 shows that this is especially true for questions generated from only a summary.

6.3.4 Performance by Number of Answer Sentences

In all conditions and metrics (P, R, F) we observe standard deviations over individual datapoints between $\pm$ 29 and $\pm$ 40 metric points. This variance can in part be explained by the varying difficulty of questions with increasing numbers of ground-truth answer sentences. The average number of ground-truth answer sentences per question lies between 5 and 6 in German and around 4 in English (Table 4.3).

The more valid choices there are, the greater the chance of retrieving correct questions, with a guaranteed precision of 100.0% if all document sentences are ground-truth answers. Achieving high recall, on the other hand, becomes increasingly difficult as the number of answer sentences increases. We show model performance as a function of number of ground-truth answer sentences exemplarily for one German model (Llama-3-70B) in Fig. 6.2. As expected, average recall becomes roughly linearly more difficult as the number increases, whereas average precision already starts high and approaches 100.0% for questions with more than 10 annotated answer sentences.

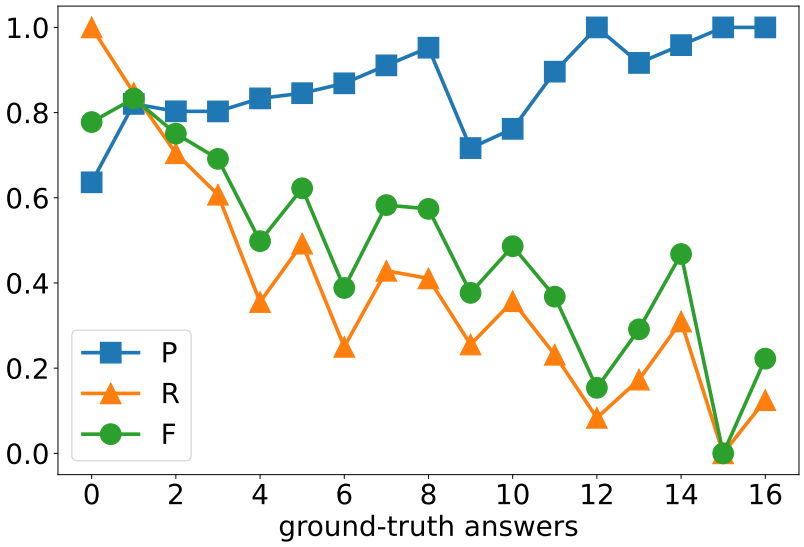

Figure 6.2: Performance of zero-shot Llama-3-70B analyzed according to the number of ground-truth answer sentences for English questions and documents.

6.4 Multilingual QA and Machine Translation

OMoS is intended as a solution in a multilingual QA setting to support newcomers and refugees in finding the information they need upon arrival in Germany. To this end, it is important to assess the multilingual capabilities of various QA approaches and therefore their suitability for this application. Hence, we conduct the same sentence extraction experiments of Section 6.3 in a multilingual setting with additional languages. We select a subset of the best performing zero-shot models (Mixtral-8x7B, Llama-3-70B) and DeBERTa. In addition, we compare those results with a retranslation setting, in which we assess the quality and performance implications of machine translating user queries. The settings and languages are described in Section 6.4.1. We then present sentence-level (Section 6.4.2) and question-level (Section 6.4.3) results.

6.4.1 Languages and Settings

We evaluate models on the following additional languages that are highly relevant in the migration context: Arabic (ar), French (fr), and Ukrainian (uk). These and other languages are more challenging due to their limited resources and much different language structure (German and English are closely related). Furthermore, Arabic and Ukrainian both use a non-Latin alphabet: The Arabic and Cyrillic alphabet. We use machine translation with DeepL to translate the question and, sentence-by-sentence, the document for each instance of the original OMoS-QA dataset.

In order to assess possible adverse effects of leveraging machine translation and to compare it to directly querying the model with the question in its original language, we evaluate the performance in an additional retranslation setting. To this end, we combine the original German documents with retranslated questions, i.e., questions that are first translated to aforementioned languages and then back to German. This corresponds to the use of machine translation in the OMoS setting, as only user input (and possibly the answers) are subject to translation, while the document corpus remains unchanged. However, questions are translated twice in the retranslation setting and results should thus be considered as lower performance boundary. Since German is the original dataset

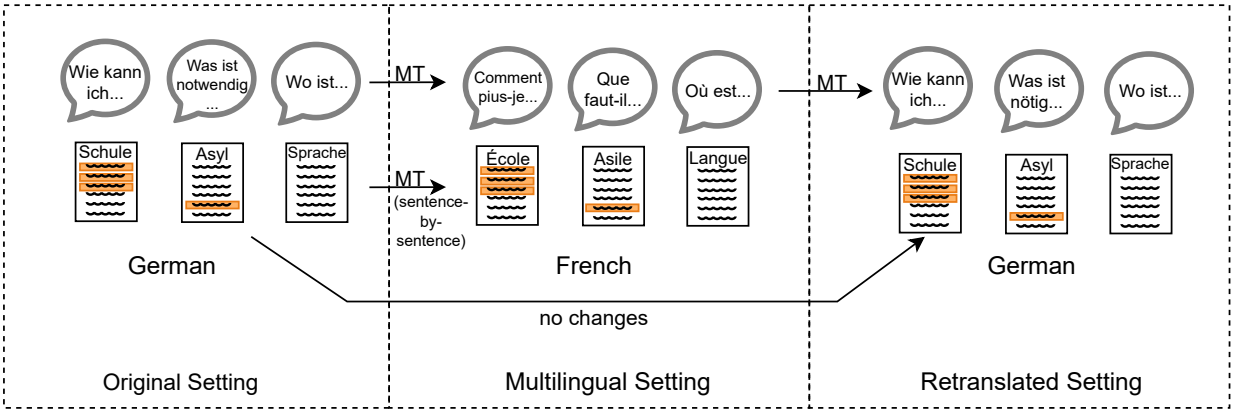

Figure 6.3: Setup of the multilingual and retranslated settings with machine translation (MT) for our multilingual experiments using the example of French. The process is analogous for Arabic and Ukrainian.

language of OMoS-QA, there are no results for the retranslated setting. Correspondingly, numbers for German are the same as in Table 6.2.

6.4.2 Sentence-Level Results

The results are shown in Table 6.3. On the left side of the table, we compare sentence-level results of different languages in both a multilingual and a retranslated setting for select models. Compared to the performances on the original German dataset version, all models display lower performance in both the multilingual and the retranslated setting for Arabic, French, and Ukrainian. Llama-3-70B shows slightly higher precision for retranslated Arabic (+0.5%) and Ukrainian (+0.1%), however, this comes at a cost of a clearer decrease in recall (−2.5% and −3.3% respectively). For the multilingual setting, French results were the closest to German. With exception to Mixtral-8x7B, the F1-score for French is at least 2% higher. Similarly, while retranslating improves F1-score performance compared to directly querying the LLM for Arabic and Ukrainian in all settings by up to +3.4%, retranslating French comes at a performance loss for Llama-3-70B and DeBERTa. Mixtral-8x7B, on the other hand, shows a performance improvement (+1.4%) for retranslating French to German, although it is explicitly advertised as "fluent in French."[1] The biggest performance loss is displayed by Llama-3-70B in the multilingual setting in Ukrainian (−5.3%) and Arabic (−4.8%).

In general, the observed performance differences are observable but not as notable as expected. This is especially the case for Arabic and Ukrainian, as the differences in the alphabet, grammar, and language origins are significant. While machine translation seems to have a slightly better performance for these languages, a performance deterioration compared to the original German dataset is still measurable. However, the questions are translated twice in our setup, and, as a consequence, the actual implications should be smaller.

[1] https://mistral.ai/technology#models

		Sentence-level Answers						Question-level Unanswerability					
		Multilingual			German Retrans.			Multilingual			German Retrans.		
Model	Lang.	P	R	F	P	R	F	P	R	F	P	R	F
Mixtral-8x7B	de	74.5	47.1	57.7	–	–	–	68.9	56.4	62.0	–	–	–
	ar	72.5	42.7	53.8	77.8	45.2	57.2	62.8	49.1	55.1	55.4	56.4	55.9
	fr	74.2	43.7	55.0	75.0	45.2	56.4	64.1	45.5	53.2	57.4	49.1	52.9
	uk	69.3	46.4	55.6	74.7	45.8	56.8	73.2	54.5	62.5	58.2	58.2	58.2
Llama-3-70B	de	**85.5**	46.6	60.3	–	–	–	69.8	67.3	68.5	–	–	–
	ar	80.9	42.2	55.5	**86.0**	44.1	58.3	71.4	54.5	61.9	61.0	65.5	63.2
	fr	84.1	44.9	58.5	84.3	43.5	57.4	72.9	63.6	68.0	63.8	**67.3**	65.5
	uk	82.4	41.3	55.0	85.6	43.3	57.5	**74.5**	63.6	**68.6**	**64.9**	**67.3**	**66.1**
DeBERTa	de	62.6	**62.4**	**62.5**	–	–	–	56.2	65.5	60.5	–	–	–
	ar	63.3	54.9	58.8	65.2	53.5	58.8	43.4	60.0	50.4	44.0	**67.3**	53.2
	fr	66.3	56.9	61.2	61.4	**59.9**	**60.6**	50.7	67.3	57.8	53.8	63.6	58.3
	uk	54.7	61.4	57.9	62.2	55.9	58.8	57.1	**72.7**	64.0	48.7	**67.3**	56.5

Table 6.3: Test set performance (in %) of zero-shot LLMs and finetuned DeBERTa on sentence-level answer extraction (left) and detection of unanswerable questions (right) for multilingual and retranslated settings. In the multilingual setting, questions and documents are machine translated to the respective language. In the retranslated setting, the question is retranslated back to German and paired with the original German document. The best result in each column is **bolded**.

6.4.3 Question-Level Unanswerability

Similarly to Section 6.3.3, we infer question-level unanswerability from sentence-level answer extraction results. If no sentence of a document is marked as answer, we treat the question as unanswerable given the document. In contrast to question-level answer extraction, the German results are not necessarily better than those of other languages in the multilingual setting, but they always outperform the retranslated results. Surprisingly, all models perform slightly better in the Ukrainian multilingual setting than on the original German dataset (up to +3.5%, DeBERTa) and mostly considerably better than on Arabic and French (up to +13.6%). Especially Ukrainian precision is high among all models, which is in line with low precision on the sentence-level, i.e., more sentences are marked as answer. Retranslating only yields small performance improvements for French for DeBERTa and for Arabic for all models. Otherwise, directly querying models leads to better question-level results (up to +7.5%).

6.5 Cross-Language QA

In the following, we evaluate a third approach to QA in this multilingual setting, in which we expect user queries in various languages: We conduct a cross-language QA experiment, i.e., we prompt a LLM with the same prompts as before, but use documents in another language than the questions. This approach has the advantage that no machine translation is needed and, at the same time, it is possible to decide on a language for which documents are available. We pilot the experiment with Llama-3-70B, as it is the LLM performing best in Section 6.3 and Section 6.4. Mixtral-8x7B, albeit exhibiting similar performance, is already used for QG in the dataset construction for question generation and thus might have a slight advantage (compare Panickssery et al. (2024)).

The results of the cross-language pilot are shown in Table 6.4. Surprisingly, asking questions in

Document	Question	Sentence-level P	Sentence-level R	Sentence-level F	Question-level P	Question-level R	Question-level F
Arabic	Arabic	80.9	42.2	55.5	71.4	54.5	61.9
Arabic	English	82.7	44.4	57.8	74.0	67.3	70.5
Arabic	German	81.9	43.1	56.4	72.7	72.7	**72.7**
English	Arabic	80.6	44.0	56.9	74.0	67.3	70.5
English	English	84.8	46.7	60.2	**74.5**	63.6	68.6
English	German	83.2	45.6	58.9	73.5	65.5	69.2
German	Arabic	84.6	41.8	56.0	63.1	**74.5**	68.3
German	English	**85.8**	**48.2**	**61.7**	70.9	70.9	70.9
German	German	85.5	46.6	60.3	69.8	67.3	68.5

Table 6.4: Test set performance (in %) of zero-shot Llama-3-70B on sentence-level answer extraction (left) and detection of unanswerable questions (right) on cross-language question-document pairs. The best result in each column is **bolded**.

Model	Method	German P	German R	German F	English P	English R	English F
Llama-3-70B	Explicit	59.0	**83.6**	**69.2**	62.3	**78.2**	**69.4**
	Inferred	69.8	67.3	68.5	74.5	63.6	68.6
DeBERTa	Explicit	**75.0**	43.6	55.2	**75.0**	54.5	63.2
	Inferred	56.2	65.5	60.5	59.4	69.1	63.9

Table 6.5: Test set performance (in %) of zero-shot Llama-3-70B and finetuned DeBERTa on explicit and inferred question-level unanswerability detection. The best result in each column is **bolded**.

different languages than the provided document does not necessarily hurt LLM performance. In fact, Llama-3-70B performed best on English questions with German documents, showing both the highest precision (85.8%) and recall (48.2%) for sentence-level answer extraction. Compared to purely English or German prompts, especially recall is improved, resulting in a 1.5% higher F1-score. However, this only holds true for English questions. German and, even more so, Arabic questions considerably worsen the results. For the document language, German shows better results than English, with Arabic also seemingly the most challenging. Thus, it seems that English questions are favorable over questions in other languages, while for documents German allows Llama-3-70B to score highest. Using Arabic both as question language or document language hurts performance, with Arabic-Arabic showing the poorest results. Results of question-level unanswerability detection are more mixed. Arabic documents lead to increased recall, i.e., detecting more unanswerable questions, while precision slightly drops. This is mostly due to increasingly invalid LLM output.

As a conclusion, we show that cross-language prompts can improve performance over same-language prompts. However, this is primarily the case for English questions and different language documents, with German questions working better than Arabic ones. Thus, since we expect questions mostly in different languages than English, cross-language prompts are not really applicable to our scenario. Nevertheless, machine translating the questions to English could be an option.

6.6 Explicit Unanswerability Detection

While the answerability of the question can be trivially inferred from the extracted answer sentences, we want to evaluate models' capabilities to explicitly detect unanswerable questions in a pilot experiment. For this purpose, we prompt Llama-3-70B to generate either the text `[YES]`, if the question is answerable, or `[NO]`, if unanswerable. We do this in a zero-shot setting, i.e., without showing the model any examples. The prompt is shown in Fig. 5.7. Additionally, we finetune DeBERTa separately on German and English QA pairs to detect unanswerable questions. The results are shown in Table 6.5. We show the results together with question unanswerability inferred from extracted answer sentences for zero-shot Llama-3-70B and DeBERTa as previously shown in Table 6.2.

Explicit unanswerability detection with zero-shot Llama-3-70B exhibits significantly lower precision (de: −10.8%; en: −12.2%) and higher recall (de: +16.3%; en: +14.9%) compared to inferred unanswerability detection. For finetuned DeBERTa, the opposite is the case: Precision is relatively high for both German and English, while recall is low, especially for German with only 43.6%. In order to detect unanswerable questions as well as possible and therefore assure high trustworthiness of our answers, higher recall is preferable over higher precision on this task. In general, explicit unanswerability detection with Llama-3-70B therefore shows the best results by scoring the highest recall and F1-score. Additional prompt engineering, for example with few-shot prompting, could be employed to further enhance model performance on this task.

Explicit detection with DeBERTa, however, performs the worst among all settings as the model seems to be reluctant to mark questions as unanswerable. These poor results might be due to the small number of unanswerable questions in the train and development partitions with a share of less than 20%. This necessity for substantial amounts of training data for the training or finetuning of models poses a limitation of encoder-only models. In contrast, LLMs can be employed without the need for any training data in the zero-shot setting or only a few instances for few-shot prompting. Another potentially influential factor for the poor performance of DeBERTa on explicitly detecting unanswerable questions is the use of truncation for long documents. The truncation causes questions classified as answerable not having any answers in the truncated document fed to the model as the answers are cut-off. The model therefore is trained on unanswerable questions classified as answerable, which possibly explains low recall in detecting unanswerable questions. In order to mitigate this issue, a sliding window could be employed in future work.

6.7 Conclusion

Our task of multilingual QA with sentence extraction is challenging for both LLM-based and binary classification approaches. We conduct several experiments to evaluate model capabilities in finding answer sentences to a question in a document and classifying whether the question is answerable using the given document at all. Prompting models to produce consistent output and finding adequate postprocessing methods is imperative for sentence extraction using LLMs. Few-shot prompting increases instruction following and model performance for most models. In contrast, few-shot prompting has strong negative implications for Mistral-7B. The model overshoots on precision by extracting considerably fewer sentences, which severely hurts model recall. Additional experiments with other few-shot samples or different compositions of answerable and unanswerable examples could be conducted to further evaluate their influence on model results. For finetuning a classifier for binary sentence classification, we determine a context window of surrounding sentences of size 3 as optimal, while smaller and bigger sizes lead to decreased performance.

In general, our models demonstrate good results on OMoS-QA, both on extracting answer sentences and detecting unanswerable questions. Our bigger open-weights models, Mixtral-8x7B and Llama-3-70B, generally show better results compared to their smaller companions and even GPT-3.5-Turbo. Answer sentence extraction has high precision exceeding 85% in some LLM and 60% in finetuned settings. Recall, on the other hand, is significantly lower with 40% to 50% for LLMs and slightly higher ranging from 55% to 70% for DeBERTa. However, this is in line with our goals of providing trustworthy answers and might be influenced by our dataset creation process. In particular, the expansion of answers described in Section 4.3.2 probably increases precision while lowering recall, with influences of the number of ground-truth answers on model performance shown in Section 6.3.4. Additionally, we have observed humans to also struggle with decision-making during the dataset construction: Deciding whether a sentence is required for an answer or just additional context and therefore not actually part of an answer has produced lots of instances with low IAA. Due to the non-factoid nature of the questions and the dataset, it is a non-trivial and challenging task to decide. However, F1-scores of some LLMs and DeBERTa still exceed the—not directly comparable—human agreement on the unfiltered dataset with up to 64.9%.

In order to detect unanswerable questions, the 5-shot setting considerably improved the amount of retrieved unanswerable questions without hurting precision too much. The best LLMs performed significantly better on this question-level task than DeBERTa. While Mixtral-8x7B and Llama-3-70B again show the best results among our open-weight models, GPT-3.5-Turbo is on par.

Both the LLMs and finetuned classifier DeBERTa perform surprisingly well even for non-Latin alphabet languages such as Arabic and Ukrainian. However, a performance gap to European languages with greater focus in research is still measurable especially for Arabic. Furthermore, applying machine translation on user questions first outperforms directly querying the model with questions in different languages. Providing LLMs with questions and documents in different languages does not cause notable adverse performance implications compared to same-language pairs. English questions and German documents even outperforms German-German and English-English question-document pairs.

A possible limitation of our experiments regarding Mixtral-8x7B is the use of this model for QG in the dataset construction process, which could cause an unfair advantage. This could be investigated in future work by evaluating models on questions generated by humans or other generative models.

7. Discussion and Outlook

Dataset Construction. Our proposed dataset construction approach produces a high-quality extractive dataset with high agreement QA pairs. Leveraging both NLP approaches and voluntary crowdsourcing greatly facilitates the construction of a dataset, even in low-resource settings regarding allowed costs and little availability of working hours. From our initial attempt of directly crowdsourcing QA pairs using a form, we conclude that the construction needs to be divided into separated and restrained steps and contributing by volunteers has to be easy and efficient.

The focus on simple and well-posed one-part questions from a single source, Mixtral-8x7B, poses a limitation to our dataset. While we try to elicit diverse questions by prompting for multiple questions per document as well as prompting with and without evidence, this might not completely reflect real-life usage of a QA system, even more so in a multilingual immigration setting. Hence, additional research is required to extend our approach to include ill-posed questions, keyword-based queries, or questions with typos and translation errors in our dataset. Ambiguous questions are an additional potential cause of problems not yet investigated.

Extending the dataset in quantity could be another angle for further improvements. A dataset consisting of 906 QA pairs is still relatively small and resulting limitations are especially visible in training or finetuning language models on less commonly observed features, such as the detection of unanswerable questions. To encourage more volunteers to annotate questions, further improvements to our annotation tool such as gamification, for example using highscores, leaderboards, or challenges, are possible.

From partly low human agreement on answer sentences and even deciding whether there is an answer at all in the unfiltered dataset (Section 4.3, Table 4.1), we conclude that QA with non-factoid questions is a challenging task and needs further research. Non-factoid questions, as opposed to factoid questions, are not answerable using simple facts and often require complex or subjective answers. While we employ careful dataset analysis and filtering to ensure high-quality instances, common patterns and the cause of annotations with low agreement or even complete disagreement regarding a question's answerability should be further investigated. A clearer definition of what constitutes an answer and what is just additional context might increase inter-annotator agreement. Low-agreement questions could be annotated additionally by domain experts.

LLM Comparison. Most LLMs exhibit high precision and medium recall on the task of answer sentence extraction (Section 6.3, Table 6.2). We interpret those results as largely positive, in particular with respect to our goal of building a reliable system that errs on the side of presenting fewer, higher precision results to the user. We have shown that SOTA open-weight LLMs such as Llama-3-70B and Mixtral-8x7B can easily compete and mostly outperform closed-weight GPT-3.5-Turbo on our dataset. Even the significantly smaller Llama-3-8B, especially in English QA, is not too far off and also exhibits high precision in extracting answer sentences. In general, few-shot prompting increases model performance and instruction following, which is especially important if a specifically formatted output is necessary to allow for useful postprocessing as in the proposed extractive QA setting.

The good performance of open-weight LLMs rises the opportunity to self-host and use LLMs with full control—to the extend possible for LLMs—without the need to pay for third-party API usage. In our sensitive application context, this additionally allows for higher privacy standards as no data has to be transferred to third-parties in- or outside the European Union. However, running

LLMs for inference is costly in hardware resources, computing power, and electricity consumption, especially for bigger models with more parameters. Few-shot Llama-3-8B might serve as a good middle ground between costs and performance. In terms of model performance of Mixtral-8x7B, it is important to keep in mind that all our questions are also generated with this model. This likely contributes to its good performance (Panickssery et al., 2024).

Finetuned Classifier. DeBERTa also shows good results on extractive QA, in particular in English with the highest F1-score among all models (Section 6.3, Table 6.2). In comparison to the best LLMs, the considerably lower precision contradicts our goals of producing trustworthy answers. To mitigate this issue, future research could be put into improving the finetuning and aiming for a higher precision while still maintaining a satisfactory recall. In general, however, the results with this finetuned classifier look very promising and show that for some NLP tasks such as extractive QA, SOTA LLMs are not necessarily required and other approaches can provide competitive or even better results. At the same time, finetuning of encoder-only models offers superior control over and interpretability of the results, whereas the only possibility to control the output of LLMs is through prompt engineering and a few hyperparameters. Furthermore, apart from initial finetuning, running these smaller models comes at a fraction of computing power and hardware requirements and results can be computed faster. Question-level unanswerability detection might be improved by applying a sliding window instead of truncating longer documents to avoid misrepresenting answerable questions by omitting relevant answer sentences.

Deciding on a solution to implement for our QA system does not necessarily have to be a binary choice between different LLMs and classifiers. Instead, a combination of different approaches is possible as well: For example, we could use a binary classifier to elicit answer sentences, possibly from multiple documents, and subsequently prompt a LLM to decide whether it is a valid answer and/or create a summarized answer for users.

Multilingual Capabilities and Machine Translation. Regardless of the chosen approach to extractive QA, we conclude that leveraging machine translation to German or English before prompting or querying the model is favorable over doing so directly without MT (Section 6.4, Table 6.3). Our cross-language experiments show that while providing a LLM with document and question in different languages does not necessarily hurt or can even improve model performance, this is only the case for English, and slightly less so, for German questions (Section 6.5, Table 6.4). Arabic questions always deteriorate the performance, and, as a consequence, cross-language prompts are not really applicable in our scenario. Apart from better performance exhibited for (retranslated) German and English QA than for other languages, also practical reasons for the actual implementation are decisive: In the Integreat-App, our real-life information corpus, documents are originally written in German and are therefore also guaranteed to be available in German. The availability of other languages differs per region. While for example Arabic, Farsi, French, or Ukrainian are fairly common in the Integreat-App and English is nearly always present, translations to some other languages are rarely or never available. Hence, without employing cross-language prompting of LLMs, some kind of translation is necessary, with translating user queries being the more efficient. Furthermore, both constructing few-shot prompts and finetuning binary classifiers has to be done individually for every language. This additional overhead for each supported language is necessary whenever changes are made to the QA system or process. In the case of finetuned LMs, this also leads to the necessity of running multiple models in parallel.

Chatbot. So far, we only investigated plain QA systems. An increasing number of today's user-faced QA systems are implemented or at least presented in a chatbot-like setting or user interface and make direct use of generative LLMs in the form of ChatGPT-like bots. However, we question whether this is a suitable approach for OMoS, as it is hindering our goals of trustworthy answers in this sensitive setting. Users of our QA systems should be able to trust the answers, reproduce the origin and reasoning behind them, and should not be faced with toxic or discriminating language. Shah and Bender (2024) make compelling arguments, e.g., possibly ungrounded answers and a lack of transparency, against the use of LLMs in important QA/IR settings and Dahl et al. (2024) show that even the latest LLMs such as GPT-4 hallucinate in at least 58% of queries about legal topics in the US, with even poorer numbers for GPT-3.5-Turbo and Llama-2. As laws and regulations in the German migration context are changed frequently and are dependent on a lot of personal characteristics of the user, e.g., age, country of origin, or acquired education, we expect at least similar numbers for the German migration context. We think that our extractive approach to QA is therefore better suited to the immigration context and exhibits promising results to move forward. In order to incorporate additional context in the user queries, combining previous messages is possible.

Outlook. Putting our results in the bigger picture of OMoS to develop a multilanguage QA system to support human migration counselors, additional factors have to be considered besides pure performance results. The QA system we investigated in this work is only a part of the intended OMoS system. It is supposed to be the first contact point of help-seeking newcomers and to provide information from the Integreat-App in an easy-to-use manner without long waiting times. However, the QA system is also intended as a filtering of user queries for the second escalation step, human counseling. To prevent human counselors from being overworked and flooded with easy-to-answer questions, users can only request direct messaging after the QA system fails to sufficiently answer the question. In order to allow for a seamless conversation, previous questions and answers of the same user should be persisted and accessible to human counselors. Additionally, it should be clear for users at which time the chat counterpart is human or machine. In further iterations, OMoS could be extended to provide comprehensive support for newcomers, including appointment booking, inclusion of documents and pictures, or portability to other devices using user accounts. For this application as an integrated QA system in combination with human counseling, we see promising results to successfully solve information needs of newcomers.

8. Conclusion

In this work, we have considered extractive QA in a German migration context. To this end, we have created a dataset tailored to this scenario, OMoS-QA. We have further conducted several experiments on extractive QA on OMoS-QA evaluating different performance of various models on different tasks, settings, and languages.

We have shown that LLMs have the capabilities to advance the construction of a dataset, exemplarily by automatically generating questions for a QA dataset. Additionally, crowdsourcing can greatly facilitate the dataset annotation process by yielding high numbers of annotations. However, it is a necessity that the human annotation task is modular and restrained to allow for an efficient and error-resistant annotation process. By applying question- or user-level filtering based on inter-annotator agreement of annotations, a high quality and high agreement dataset can be constructed, even if annotators are untrained volunteers. We have created OMoS-QA in a modular step-by-step approach leveraging both LLMs and voluntary crowdsourcing. To this end, we have developed a custom web-based annotation tool to foster the human annotation process. Our extractive dataset consists of 906 high-quality QA pairs in German and English.

We have focused on extractive QA to ensure answers are trustworthy in this highly sensitive migration context. Both LLMs and finetuned classifiers (DeBERTa) have shown good results on sentence-level answer extraction on our OMoS-QA dataset. The LLMs have generally exhibited high precision and medium recall, with Llama-3-70B and Mixtral-8x7B performing best in most experiments. We have therefore shown that the latest open-weight models (as of writing this thesis, i.e., July 2024) can compete with and even outperform closed-source GPT-3.5-Turbo on our task. Few-shot prompting has usually improved model performance, although it has also led to an adverse effect for some smaller models. Several models have surpassed the human agreement. Finetuned DeBERTa has provided more balanced and stable results, i.e., lower precision but higher recall compared to LLMs. The models have performed surprisingly well on machine-translated multilingual questions and documents, though there has still been a performance loss compared to the original dataset. Automatic retranslating to German has partly mitigated this issue. In our cross-language pilot experiment, the model has performed worse with Arabic questions, and, surprisingly, better on cross-language prompts with English questions compared to same-language prompts.

Apart from using our models to elicit answer sentences to questions, we have experimented with classifying whether a question is answerable or not, given a document. We have shown that both LLMs and DeBERTa implicitly detect most unanswerable questions by not extracting any answer sentences. Explicit unanswerability detection has only increased performance for LLM prompting. For DeBERTa, explicit unanswerability detection has led to worsened results.

Limitations. We only consider extractive QA on already provided documents. In an actual application scenario, document retrieval among the complete document collection needs to be implemented as preceding step to the QA system discussed in this work. Additionally, the questions in our OMoS-QA dataset stem from a single source, Mixtral-8x7B, and are exclusively simple German and English one-part questions. However, we try to elicit diverse questions, e.g., using different prompts. We neither consider keyword-based user queries nor complicated or incorrect questions, i.e., questions containing typos or poor grammar. Furthermore, we do not explicitly consider ambiguous questions and only generate and annotate questions in German or English. This poses a limitation, as immigrants arriving in Germany are from all around the world, and often lack

German and English language skills. In some of our experiments, we employ Mixtral-8x7B, which is already used to generate questions. This likely improves model performance as LLMs seem to favor their own output (Panickssery et al., 2024).

List of Figures

List of Tables

Bibliography

Chris Alberti, Daniel Andor, Emily Pitler, Jacob Devlin, and Michael Collins. 2019. Synthetic QA corpora generation with roundtrip consistency. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 6168–6173, Florence, Italy. Association for Computational Linguistics.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. MS MARCO: A human generated machine reading comprehension dataset. ArXiv Preprint 1611.09268.

Max Bartolo, Tristan Thrush, Robin Jia, Sebastian Riedel, Pontus Stenetorp, and Douwe Kiela. 2021. Improving question answering model robustness with synthetic adversarial data generation. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8830–8848, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Frederic Bechet, Elie Antoine, Jérémy Auguste, and Géraldine Damnati. 2022. Question generation and answering for exploring digital humanities collections. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 4561–4568, Marseille, France. European Language Resources Association.

Jeroen Bertels, Tom Eelbode, Maxim Berman, Dirk Vandermeulen, Frederik Maes, Raf Bisschops, and Matthew B. Blaschko. 2019. *Optimizing the Dice Score and Jaccard Index for Medical Image Segmentation: Theory and Practice*, page 92–100. Springer International Publishing.

Quentin Brabant, Gwénolé Lecorvé, and Lina M. Rojas Barahona. 2022. CoQAR: Question rewriting on CoQA. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 119–126, Marseille, France. European Language Resources Association.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. ArXiv Preprint 2005.14165.

Sébastien Bubeck, Varun Chandrasekaran, Ronen Eldan, Johannes Gehrke, Eric Horvitz, Ece Kamar, Peter Lee, Yin Tat Lee, Yuanzhi Li, Scott Lundberg, Harsha Nori, Hamid Palangi, Marco Tulio Ribeiro, and Yi Zhang. 2023. Sparks of artificial general intelligence: Early experiments with GPT-4. ArXiv Preprint 2303.12712.

Shreya Chandrasekhar, Chieh-Yang Huang, and Ting-Hao Huang. 2023. Good data, large data, or no data? comparing three approaches in developing research aspect classifiers for biomedical papers. In *The 22nd Workshop on Biomedical Natural Language Processing and BioNLP Shared Tasks*, pages 103–113, Toronto, Canada. Association for Computational Linguistics.

Mark Chen, Jerry Tworek, Heewoo Jun, Qiming Yuan, Henrique Ponde de Oliveira Pinto, Jared Kaplan, Harri Edwards, Yuri Burda, Nicholas Joseph, Greg Brockman, Alex Ray, Raul Puri,

Gretchen Krueger, Michael Petrov, Heidy Khlaaf, Girish Sastry, Pamela Mishkin, Brooke Chan, Scott Gray, Nick Ryder, Mikhail Pavlov, Alethea Power, Lukasz Kaiser, Mohammad Bavarian, Clemens Winter, Philippe Tillet, Felipe Petroski Such, Dave Cummings, Matthias Plappert, Fotios Chantzis, Elizabeth Barnes, Ariel Herbert-Voss, William Hebgen Guss, Alex Nichol, Alex Paino, Nikolas Tezak, Jie Tang, Igor Babuschkin, Suchir Balaji, Shantanu Jain, William Saunders, Christopher Hesse, Andrew N. Carr, Jan Leike, Josh Achiam, Vedant Misra, Evan Morikawa, Alec Radford, Matthew Knight, Miles Brundage, Mira Murati, Katie Mayer, Peter Welinder, Bob McGrew, Dario Amodei, Sam McCandlish, Ilya Sutskever, and Wojciech Zaremba. 2021. Evaluating large language models trained on code. ArXiv Preprint 2107.03374.

Peter Christen, David J. Hand, and Nishadi Kirielle. 2023. A review of the F-measure: Its history, properties, criticism, and alternatives. *ACM Comput. Surv.*, 56(3).

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855, Melbourne, Australia. Association for Computational Linguistics.

Elizabeth Clark, Tal August, Sofia Serrano, Nikita Haduong, Suchin Gururangan, and Noah A. Smith. 2021. All that's 'human' is not gold: Evaluating human evaluation of generated text. 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 7282–7296, Online. Association for Computational Linguistics.

Matthew Dahl, Varun Magesh, Mirac Suzgun, and Daniel E Ho. 2024. Large legal fictions: Profiling legal hallucinations in large language models. *Journal of Legal Analysis*, 16(1):64–93.

Vila-Suero Daniel and Aranda Francisco. 2023. Argilla - Open-source framework for data-centric NLP.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Martin d'Hoffschmidt, Wacim Belblidia, Quentin Heinrich, Tom Brendlé, and Maxime Vidal. 2020. FQuAD: French question answering dataset. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1193–1208, Online. Association for Computational Linguistics.

Liam Dugan, Eleni Miltsakaki, Shriyash Upadhyay, Etan Ginsberg, Hannah Gonzalez, DaHyeon Choi, Chuning Yuan, and Chris Callison-Burch. 2022. A feasibility study of answer-agnostic question generation for education. In *Findings of the Association for Computational Linguistics: ACL 2022*, pages 1919–1926, Dublin, Ireland. Association for Computational Linguistics.

Kelvin Han, Thiago Castro Ferreira, and Claire Gardent. 2022. Generating questions from Wikidata triples. In *Proceedings of the Thirteenth Language Resources and Evaluation Conference*, pages 277–290, Marseille, France. European Language Resources Association.

Luheng He, Mike Lewis, and Luke Zettlemoyer. 2015. Question-answer driven semantic role labeling: Using natural language to annotate natural language. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 643–653, Lisbon, Portugal. Association for Computational Linguistics.

Pengcheng He, Jianfeng Gao, and Weizhu Chen. 2023. DeBERTaV3: Improving DeBERTa using ELECTRA-style pre-training with gradient-disentangled embedding sharing. ArXiv Preprint 2111.09543.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. 2021. DeBERTa: Decoding-enhanced BERT with disentangled attention. ArXiv Preprint 2006.03654.

Sophie Henning, Talita Anthonio, Wei Zhou, Heike Adel, Mohsen Mesgar, and Annemarie Friedrich. 2023. Is the answer in the text? challenging ChatGPT with evidence retrieval from instructive text. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 14229–14241, Singapore. Association for Computational Linguistics.

International Organization for Migration IOM. 2023. DTM Ukraine - internal displacement report - general population survey round 13 (11 May - 14 June 2023).

Albert Q. Jiang, Alexandre Sablayrolles, Arthur Mensch, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Florian Bressand, Gianna Lengyel, Guillaume Lample, Lucile Saulnier, Lélio Renard Lavaud, Marie-Anne Lachaux, Pierre Stock, Teven Le Scao, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2023. Mistral 7b. ArXiv Preprint 2310.06825.

Albert Q. Jiang, Alexandre Sablayrolles, Antoine Roux, Arthur Mensch, Blanche Savary, Chris Bamford, Devendra Singh Chaplot, Diego de las Casas, Emma Bou Hanna, Florian Bressand, Gianna Lengyel, Guillaume Bour, Guillaume Lample, Lélio Renard Lavaud, Lucile Saulnier, Marie-Anne Lachaux, Pierre Stock, Sandeep Subramanian, Sophia Yang, Szymon Antoniak, Teven Le Scao, Théophile Gervet, Thibaut Lavril, Thomas Wang, Timothée Lacroix, and William El Sayed. 2024. Mixtral of experts. ArXiv Preprint 2401.04088.

Dan Jurafsky and James H. Martin. 2023. *Speech and Language Processing*. Prentice Hall.

Steffen Kleinle, Jakob Prange, and Annemarie Friedrich. 2024. OMoS-QA: A dataset for cross-lingual extractive question answering in a german migration context. In *Proceedings of the 20th Conference on Natural Language Processing (KONVENS 2024)*, Vienna, Austria. To appear.

Jan-Christoph Klie, Michael Bugert, Beto Boullosa, Richard Eckart de Castilho, and Iryna Gurevych. 2018. The INCEpTION platform: Machine-assisted and knowledge-oriented interactive annotation. In *Proceedings of the 27th International Conference on Computational Linguistics: System Demonstrations*, pages 5–9. Association for Computational Linguistics. Event Title: The 27th International Conference on Computational Linguistics (COLING 2018).

Zsoka Koczan, Giovanni Peri, Magali Pinat, and Dmitriy Rozhkov. 2021. Migration. In *How to Achieve Inclusive Growth*. Oxford University Press.

Vishwajeet Kumar, Nitish Joshi, Arijit Mukherjee, Ganesh Ramakrishnan, and Preethi Jyothi. 2019. Cross-lingual training for automatic question generation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4863–4872, Florence, Italy. Association for Computational Linguistics.

Yinheng Li, Shaofei Wang, Han Ding, and Hang Chen. 2023. Large language models in finance: A survey. In *Proceedings of the Fourth ACM International Conference on AI in Finance*, ICAIF '23, page 374–382, New York, NY, USA. Association for Computing Machinery.

Seungyoung Lim, Myungji Kim, and Jooyoul Lee. 2019. KorQuAD1.0: Korean QA dataset for machine reading comprehension. ArXiv Preprint 1909.07005.

Nelson F. Liu, Kevin Lin, John Hewitt, Ashwin Paranjape, Michele Bevilacqua, Fabio Petroni, and Percy Liang. 2024. Lost in the middle: How language models use long contexts. *Transactions of the Association for Computational Linguistics*, 12:157–173.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized BERT pretraining approach. ArXiv Preprint 1907.11692.

Man Luo, Kazuma Hashimoto, Semih Yavuz, Zhiwei Liu, Chitta Baral, and Yingbo Zhou. 2022. Choose your QA model wisely: A systematic study of generative and extractive readers for question answering. In *Proceedings of the 1st Workshop on Semiparametric Methods in NLP: Decoupling Logic from Knowledge*, pages 7–22, Dublin, Ireland and Online. Association for Computational Linguistics.

Andrew M Mcnutt, Chenglong Wang, Robert A Deline, and Steven M. Drucker. 2023. On the design of AI-powered code assistants for notebooks. In *Proceedings of the 2023 CHI Conference on Human Factors in Computing Systems*, CHI '23, New York, NY, USA. Association for Computing Machinery.

Timo Möller, Julian Risch, and Malte Pietsch. 2021. GermanQuAD and GermanDPR: Improving non-English question answering and passage retrieval. In *Proceedings of the 3rd Workshop on Machine Reading for Question Answering*, pages 42–50, Punta Cana, Dominican Republic. Association for Computational Linguistics.

Hiroki Nakayama, Takahiro Kubo, Junya Kamura, Yasufumi Taniguchi, and Xu Liang. 2018. doccano: Text annotation tool for human. Software available from https://github.com/doccano/doccano.

Isabella Olariu, Cedric Lothritz, Jacques Klein, Tegawendé Bissyandé, Siwen Guo, and Shohreh Haddadan. 2023. Evaluating parameter-efficient finetuning approaches for pre-trained models on the financial domain. In *Findings of the Association for Computational Linguistics: EMNLP 2023*, pages 15482–15491, Singapore. Association for Computational Linguistics.

OpenAI, Josh Achiam, Steven Adler, Sandhini Agarwal, Lama Ahmad, Ilge Akkaya, Florencia Leoni Aleman, Diogo Almeida, Janko Altenschmidt, Sam Altman, Shyamal Anadkat, Red Avila, Igor Babuschkin, Suchir Balaji, Valerie Balcom, Paul Baltescu, Haiming Bao, Mohammad Bavarian, Jeff Belgum, Irwan Bello, Jake Berdine, Gabriel Bernadett-Shapiro, Christopher Berner, Lenny Bogdonoff, Oleg Boiko, Madelaine Boyd, Anna-Luisa Brakman, Greg Brockman, Tim Brooks, Miles Brundage, Kevin Button, Trevor Cai, Rosie Campbell, Andrew Cann, Brittany Carey, Chelsea Carlson, Rory Carmichael, Brooke Chan, Che Chang, Fotis Chantzis, Derek Chen, Sully Chen, Ruby Chen, Jason Chen, Mark Chen, Ben Chess, Chester Cho, Casey Chu, Hyung Won Chung, Dave Cummings, Jeremiah Currier, Yunxing Dai, Cory Decareaux, Thomas Degry, Noah Deutsch, Damien Deville, Arka Dhar, David Dohan, Steve Dowling, Sheila Dunning, Adrien Ecoffet, Atty Eleti, Tyna Eloundou, David Farhi, Liam Fedus, Niko Felix, Simón Posada Fishman, Juston Forte, Isabella Fulford, Leo Gao, Elie Georges, Christian Gibson, Vik Goel, Tarun Gogineni, Gabriel Goh, Rapha Gontijo-Lopes, Jonathan Gordon, Morgan Grafstein, Scott Gray, Ryan Greene, Joshua Gross, Shixiang Shane Gu, Yufei Guo, Chris Hallacy, Jesse Han, Jeff Harris, Yuchen He, Mike Heaton, Johannes Heidecke, Chris Hesse, Alan Hickey, Wade Hickey, Peter Hoeschele, Brandon Houghton, Kenny Hsu, Shengli Hu, Xin Hu, Joost Huizinga, Shantanu Jain, Shawn Jain, Joanne Jang, Angela Jiang, Roger Jiang, Haozhun Jin, Denny Jin, Shino Jomoto, Billie Jonn, Heewoo Jun, Tomer Kaftan, Łukasz Kaiser, Ali Kamali, Ingmar Kanitscheider, Nitish Shirish Keskar, Tabarak Khan, Logan Kilpatrick, Jong Wook

Kim, Christina Kim, Yongjik Kim, Jan Hendrik Kirchner, Jamie Kiros, Matt Knight, Daniel Kokotajlo, Łukasz Kondraciuk, Andrew Kondrich, Aris Konstantinidis, Kyle Kosic, Gretchen Krueger, Vishal Kuo, Michael Lampe, Ikai Lan, Teddy Lee, Jan Leike, Jade Leung, Daniel Levy, Chak Ming Li, Rachel Lim, Molly Lin, Stephanie Lin, Mateusz Litwin, Theresa Lopez, Ryan Lowe, Patricia Lue, Anna Makanju, Kim Malfacini, Sam Manning, Todor Markov, Yaniv Markovski, Bianca Martin, Katie Mayer, Andrew Mayne, Bob McGrew, Scott Mayer McKinney, Christine McLeavey, Paul McMillan, Jake McNeil, David Medina, Aalok Mehta, Jacob Menick, Luke Metz, Andrey Mishchenko, Pamela Mishkin, Vinnie Monaco, Evan Morikawa, Daniel Mossing, Tong Mu, Mira Murati, Oleg Murk, David Mély, Ashvin Nair, Reiichiro Nakano, Rajeev Nayak, Arvind Neelakantan, Richard Ngo, Hyeonwoo Noh, Long Ouyang, Cullen O'Keefe, Jakub Pachocki, Alex Paino, Joe Palermo, Ashley Pantuliano, Giambattista Parascandolo, Joel Parish, Emy Parparita, Alex Passos, Mikhail Pavlov, Andrew Peng, Adam Perelman, Filipe de Avila Belbute Peres, Michael Petrov, Henrique Ponde de Oliveira Pinto, Michael, Pokorny, Michelle Pokrass, Vitchyr H. Pong, Tolly Powell, Alethea Power, Boris Power, Elizabeth Proehl, Raul Puri, Alec Radford, Jack Rae, Aditya Ramesh, Cameron Raymond, Francis Real, Kendra Rimbach, Carl Ross, Bob Rotsted, Henri Roussez, Nick Ryder, Mario Saltarelli, Ted Sanders, Shibani Santurkar, Girish Sastry, Heather Schmidt, David Schnurr, John Schulman, Daniel Selsam, Kyla Sheppard, Toki Sherbakov, Jessica Shieh, Sarah Shoker, Pranav Shyam, Szymon Sidor, Eric Sigler, Maddie Simens, Jordan Sitkin, Katarina Slama, Ian Sohl, Benjamin Sokolowsky, Yang Song, Natalie Staudacher, Felipe Petroski Such, Natalie Summers, Ilya Sutskever, Jie Tang, Nikolas Tezak, Madeleine B. Thompson, Phil Tillet, Amin Tootoonchian, Elizabeth Tseng, Preston Tuggle, Nick Turley, Jerry Tworek, Juan Felipe Cerón Uribe, Andrea Vallone, Arun Vijayvergiya, Chelsea Voss, Carroll Wainwright, Justin Jay Wang, Alvin Wang, Ben Wang, Jonathan Ward, Jason Wei, CJ Weinmann, Akila Welihinda, Peter Welinder, Jiayi Weng, Lilian Weng, Matt Wiethoff, Dave Willner, Clemens Winter, Samuel Wolrich, Hannah Wong, Lauren Workman, Sherwin Wu, Jeff Wu, Michael Wu, Kai Xiao, Tao Xu, Sarah Yoo, Kevin Yu, Qiming Yuan, Wojciech Zaremba, Rowan Zellers, Chong Zhang, Marvin Zhang, Shengjia Zhao, Tianhao Zheng, Juntang Zhuang, William Zhuk, and Barret Zoph. 2024. GPT-4 technical report. ArXiv Preprint 2303.08774.

Juri Opitz. 2024. A Closer Look at Classification Evaluation Metrics and a Critical Reflection of Common Evaluation Practice. *Transactions of the Association for Computational Linguistics*, 12:820–836.

Long Ouyang, Jeff Wu, Xu Jiang, Diogo Almeida, Carroll L. Wainwright, Pamela Mishkin, Chong Zhang, Sandhini Agarwal, Katarina Slama, Alex Ray, John Schulman, Jacob Hilton, Fraser Kelton, Luke Miller, Maddie Simens, Amanda Askell, Peter Welinder, Paul Christiano, Jan Leike, and Ryan Lowe. 2022. Training language models to follow instructions with human feedback. ArXiv Preprint 2203.02155.

Arjun Panickssery, Samuel R. Bowman, and Shi Feng. 2024. Llm evaluators recognize and favor their own generations. ArXiv Preprint 2404.13076.

Sameer Pradhan and Sandra Kuebler, editors. 2022. *Proceedings of the 16th Linguistic Annotation Workshop (LAW-XVI) within LREC2022*. European Language Resources Association, Marseille, France.

Archiki Prasad, Trung Bui, Seunghyun Yoon, Hanieh Deilamsalehy, Franck Dernoncourt, and Mohit Bansal. 2023. MeetingQA: Extractive question-answering on meeting transcripts. In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 15000–15025, Toronto, Canada. Association for Computational Linguistics.

Raul Puri, Ryan Spring, Mohammad Shoeybi, Mostofa Patwary, and Bryan Catanzaro. 2020. Training question answering models from synthetic data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5811–5826, Online. Association for Computational Linguistics.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training. Informal Report.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners. Informal Report.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. 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. Association for Computational Linguistics.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. 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. Association for Computational Linguistics.

Stephen E. Robertson, Steve Walker, Susan Jones, Micheline Hancock-Beaulieu, and Mike Gatford. 1994. Okapi at TREC-3. In *Text Retrieval Conference*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2018. Bidirectional attention flow for machine comprehension. ArXiv Preprint 1611.01603.

Chirag Shah and Emily M. Bender. 2024. Envisioning information access systems: What makes for good tools and a healthy web? *ACM Trans. Web*, 18(3).

Simone Tedeschi, Johan Bos, Thierry Declerck, Jan Hajič, Daniel Hershcovich, Eduard Hovy, Alexander Koller, Simon Krek, Steven Schockaert, Rico Sennrich, Ekaterina Shutova, and Roberto Navigli. 2023. What's the meaning of superhuman performance in today's NLU? In *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 12471–12491, Toronto, Canada. Association for Computational Linguistics.

Arun James Thirunavukarasu, Darren Shu Jeng Ting, Kabilan Elangovan, Laura Gutierrez, Ting Fang Tan, and Daniel Shu Wei Ting. 2023. Large language models in medicine. *Nature medicine*, 29(8):1930–1940.

Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, Aurelien Rodriguez, Armand Joulin, Edouard Grave, and Guillaume Lample. 2023a. LLaMA: Open and efficient foundation language models. ArXiv Preprint 2302.13971.

Hugo Touvron, Louis Martin, Kevin Stone, Peter Albert, Amjad Almahairi, Yasmine Babaei, Nikolay Bashlykov, Soumya Batra, Prajjwal Bhargava, Shruti Bhosale, Dan Bikel, Lukas Blecher, Cristian Canton Ferrer, Moya Chen, Guillem Cucurull, David Esiobu, Jude Fernandes, Jeremy Fu, Wenyin Fu, Brian Fuller, Cynthia Gao, Vedanuj Goswami, Naman Goyal, Anthony Hartshorn, Saghar Hosseini, Rui Hou, Hakan Inan, Marcin Kardas, Viktor Kerkez, Madian Khabsa, Isabel Kloumann, Artem Korenev, Punit Singh Koura, Marie-Anne Lachaux, Thibaut Lavril, Jenya Lee, Diana Liskovich, Yinghai Lu, Yuning Mao, Xavier Martinet, Todor Mihaylov, Pushkar Mishra, Igor Molybog, Yixin Nie, Andrew Poulton, Jeremy Reizenstein, Rashi Rungta,

Kalyan Saladi, Alan Schelten, Ruan Silva, Eric Michael Smith, Ranjan Subramanian, Xiaoqing Ellen Tan, Binh Tang, Ross Taylor, Adina Williams, Jian Xiang Kuan, Puxin Xu, Zheng Yan, Iliyan Zarov, Yuchen Zhang, Angela Fan, Melanie Kambadur, Sharan Narang, Aurelien Rodriguez, Robert Stojnic, Sergey Edunov, and Thomas Scialom. 2023b. Llama 2: Open foundation and fine-tuned chat models. ArXiv Preprint 2307.09288.

High Commissioner for Refugees United Nations. 2024. Global trends: Forced displacement in 2023.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc.

Alex Wang, Yada Pruksachatkun, Nikita Nangia, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2019a. SuperGLUE: A stickier benchmark for general-purpose language understanding systems. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355, Brussels, Belgium. Association for Computational Linguistics.

Hai Wang, Dian Yu, Kai Sun, Jianshu Chen, Dong Yu, David McAllester, and Dan Roth. 2019b. Evidence sentence extraction for machine reading comprehension. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 696–707, Hong Kong, China. Association for Computational Linguistics.

Blaise Agüera y Arcas. 2022. Do large language models understand us? *Daedalus*, 151(2):183–197.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Xingdi Yuan, Tong Wang, Yen-Hsiang Wang, Emery Fine, Rania Abdelghani, Hélène Sauzéon, and Pierre-Yves Oudeyer. 2023. Selecting better samples from pre-trained LLMs: A case study on question generation. In *Findings of the Association for Computational Linguistics: ACL 2023*, pages 12952–12965, Toronto, Canada. Association for Computational Linguistics.

Yutao Zhu, Huaying Yuan, Shuting Wang, Jiongnan Liu, Wenhan Liu, Chenlong Deng, Haonan Chen, Zhicheng Dou, and Ji-Rong Wen. 2024. Large language models for information retrieval: A survey. ArXiv Preprint 2308.07107.

A. Relationship of F1 and Jaccard

The F1 score or Dice coefficient is defined as follows:

$$\begin{aligned} F1(A,B) &= \frac{2|A \cap B|}{|A| + |B|} \\ &= \frac{2|A \cap B|}{(|A \cap B| + |A \setminus B|) + (|B \cap A| + |B \setminus A|)} \\ &= \frac{|A \cap B|}{|A \cap B| + \frac{1}{2}|A \setminus B| + \frac{1}{2}|B \setminus A|} = \frac{|A \cap B|}{G(A,B)} \end{aligned} \tag{A.1}$$

with $G(A,B) = |A \cap B| + \frac{1}{2}|A \setminus B| + \frac{1}{2}|B \setminus A|$.

$$\begin{aligned} J(A,B) &= \frac{|A \cap B|}{|A \cap B| + |A \setminus B| + |B \setminus A|} \\ &= \frac{|A \cap B|}{G(A,B)} \frac{G(A,B)}{|A \cap B| + |A \setminus B| + |B \setminus A|} \\ &= F1(A,B) \left(\frac{|A \cap B| + |A \setminus B| + |B \setminus A|}{G(A,B)} \right)^{-1} \\ &= F1(A,B) \left(\frac{2|A \cap B| - |A \cap B| + |A \setminus B| + |B \setminus A|}{G(A,B)} \right)^{-1} \\ &= F1(A,B) \left(\frac{2\left(|A \cap B| + \frac{1}{2}|A \setminus B| + \frac{1}{2}|B \setminus A|\right)}{G(A,B)} - \frac{|A \cap B|}{G(A,B)} \right)^{-1} \\ &= F1(A,B) \left(\frac{2G(A,B)}{G(A,B)} - F1(A,B) \right)^{-1} \\ &= \frac{F1(A,B)}{2 - F1(A,B)} \end{aligned} \tag{A.2}$$

Since $0 \leq F1(A,B) \leq 1$ the following holds true:

$$J(A,B) = \frac{F1(A,B)}{2 - F1(A,B)} \leq F1(A,B) \tag{A.3}$$

Since $0 \leq F1(A,B) \leq 1$ the following holds true:

$$F1(A,B) = \frac{F1(A,B)}{2 - F1(A,B)} \leq F1(A,B) \tag{A.4}$$

and for $F1(A,B) \neq 0$ and $F1(A,B) \neq 1$ even:

$$J(A,B) < F1(A,B) \tag{A.5}$$

Analogous, we can describe the F1-score through the Jaccard index:

$$F1(A,B) = \frac{2J(A,B)}{1 + J(A,B)} \tag{A.6}$$

B. Question Auditing

Audited	Original
Was bietet das JiBB an?	Was bietet das JiBB an (außer Klärung des richtigen Ansprechpartners)?
Welche Klassenstufen werden am Gymnasium unterrichtet?	Welche Klassenstufen werden am Wilhelm-Hausenstein-Gymnasium unterrichtet?
Wie kann man die Polizei anrufen?	Welche Nummer wählt man bei der Polizei?
Was sollte man vor dem Unterschreiben eines Vertrages nicht tun?	Was sollte man laut Text nicht tun, bevor man einen Vertrag unterschreibt?
Was bedeutet "Inklusion"?	Was bedeutet "Inklusion" laut dem Text?
Who can attend the German courses for initial language orientation?	Who can attend these German courses?
When is FamAra available?	When is the service available?
What emergency numbers are available?	What are the emergency numbers provided?
What type of climbing groups are offered at IG Klettern?	What type of climbing groups are offered?
How can I get a discount on medicine?	How can I get over-the-counter medication at a reduced price? (2, 3, 4, 5)
What does inclusion mean?	What does inclusion mean according to the text?
What is the age range for the school for vocational integration?	What is the age range of people who can attend this school?
What types of permanent residence permits are there?	What are the two types of permanent residence permits mentioned

Table B.1: Examples of audited questions. We manually audit and adjust the generated questions. We fix typos and rewrite in-context questions to out-of-context questions.

C. Question Answering Prompt Template

C.1 Text Extraction Prompt

As mentioned in Section 5.3, we mostly follow the prompt template proposed by Henning et al. (2023) for both our zero-shot and 5-shot experiments. We use the chunked samples shown in Fig. C.2 and their sentence-by-sentence translations to German for the 5-shot experiments. For each sample, we insert the 0-shot prompt with the chunked sample document and question as user message followed by an assistant message including the expected output.

C.2 Previous Text Extraction Prompt Iterations

Before settling on the previously described prompt, we tried a less structured approach shown in Fig. C.1.

```
Given the question and context below, find the answer sentences to the question
in the context.
Please use the format of:
## Answer: {answer} ## Sentence numbers: {answer sentence numbers}
If there is no answer in the context, use the format of:
## Answer: None. ## Numbers: -1

Question: {question}

Context: {context}
```

Figure C.1: Initial answer sentence extraction prompt.

Question 1: What do you need to open a bank account?
Document 1:
[9] When can I start learning to drive?
[10] In Germany, you may only drive a car with a valid driver's license.
[11] Beforehand, you have to attend a driving school and take theoretical and practical lessons, which you also have to pay for.
[12] You can get information about this at the driving school.
[13] When can I open my own bank account?
Answer 1: []

Question 2: What is a fictitious certificate?
Document 2:
[0] Residence with fictitious certificate
[1] Departure with a fictitious certificate
[2] With a fictitious certificate, you have a temporary right of residence.
[3] There are different types of fictitious certificate.
[4]Please note:
[5] Re-entry into the federal territory is only possible with a fictitious certificate in accordance with § 81 para.4 AufenthG possible.
Answer 2: [2]

Question 3: Where can I find information on admission procedures at vocational schools?
Document 3:
[11] Initial vocational training is possible at vocational schools and vocational colleges.
[12] Training can take place both in the dual system (training company and vocational school) or purely school-based training (vocational schools).
[13] The dates and registration requirements vary from vocational school to vocational school.
[14] Information evenings are held at vocational schools every year before enrollment.
[15] Information on the admission procedure at the vocational schools can be obtained directly from the respective school.
[5] Re-entry into the federal territory is only possible with a fictitious certificate in accordance with § 81 para.4 AufenthG possible.
Answer 3: [14, 15]

Question 4: What types of school are there in Germany?
Document 4:
[0] Support with school or personal problems
[1] Does your child need help with problems?
[2] Then these places will help you:
[3] Youth social work (JaS for short) and youth work at schools (JA for short) for school, personal or family problems:
[4] It is best to contact the school directly or the Augsburg District Office for general information:
Answer 4: []

Question 5: What topics are covered in the initial orientation courses?
Document 5:
[2] The German courses for initial language orientation (also known as initial orientation courses) teach both basic German language skills and information about life in Germany.
[3] They are a practical starting aid in the new living environment and make everyday life easier.
[4] A course comprises 300 teaching units of 45 minutes each and covers topics such as "Health/medical care", "Work", "Kindergarten/school", "Housing", "Local orientation/transport/mobility."
[5] The focus is on oral communication: participants should learn as quickly as possible to find their way around in everyday life.
[6] Across all modules, initial orientation courses are also about teaching values.
Answer 5: [2, 4, 5, 6]

Figure C.2: Chunked samples for 5-shot experiments.

D. Annotation Tool

Fig. D.1 shows the initial landing page of the annotation tool described in Section 4.2.2. Users can find additional information in regard to the task and background. Before being able to annotate, users have to agree to the processing and publication of their annotations as well as to the use thereof for machine learning.

Fig. D.2 shows the Kotlin algorithm deciding on the next question to show to users.

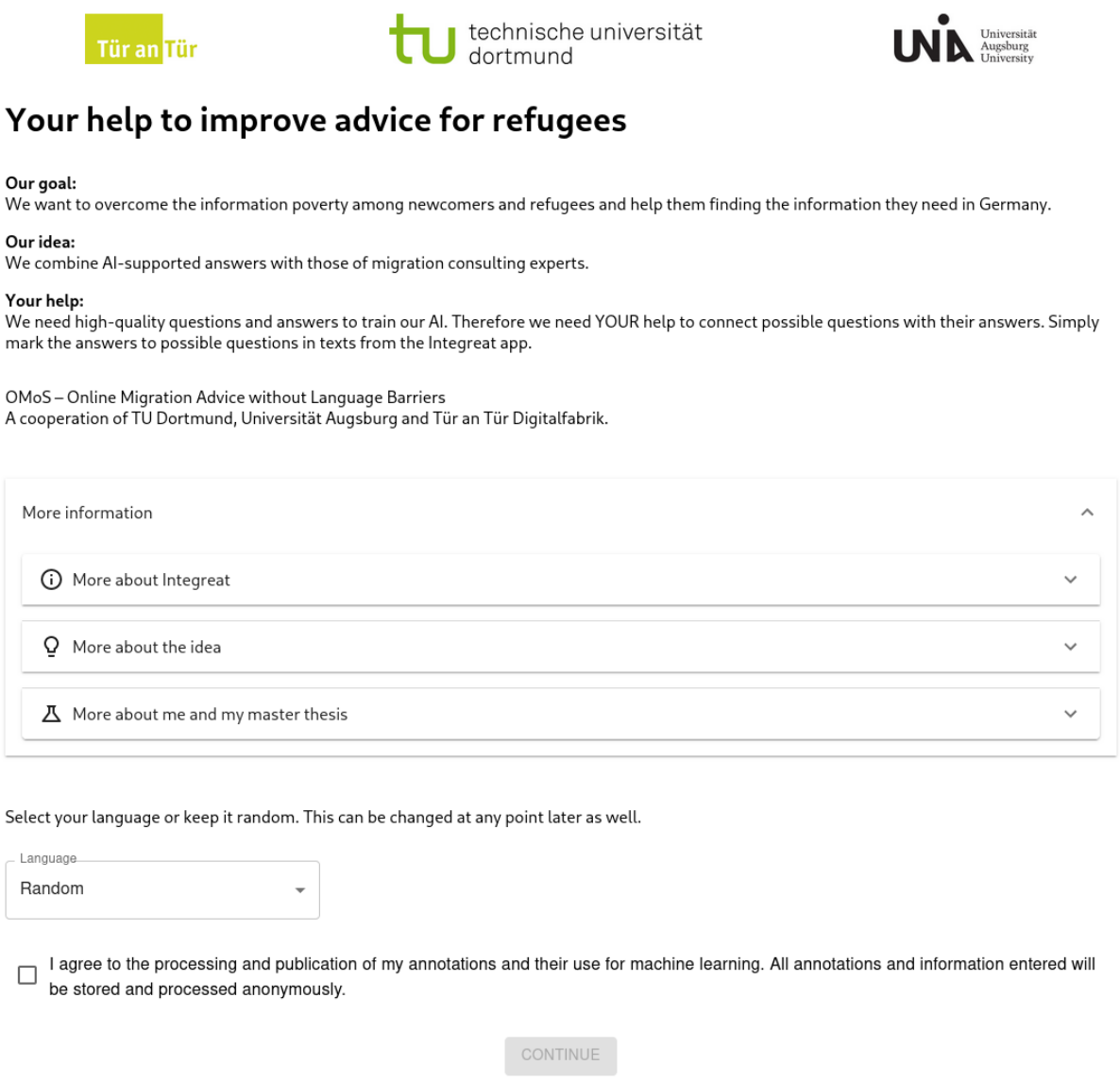

Figure D.1: Landing page of the custom annotation tool.

```kotlin
const val MAX_ANNOTATIONS_PER_QUESTION = 2
const val MAX_SINGLE = 10

fun getNextQuestion(user: String): Question? = transaction {
    val query = (Questions leftJoin Annotations)
        .slice(Questions.columns)
        .select {
            // Exclude questions the user already annotated
            notExists(Annotations.select {
                (Annotations.user eq user) and
                (Annotations.questionId eq Questions.id)
            }) and
            // Exclude archived questions
            (Questions.archived eq false) and
            // Exclude questions with enough annotations already
            (Questions.annotationCount less MAX_ANNOTATIONS_PER_QUESTION)
        }
        .groupBy(Questions.id)

    // Questions.annotationCount excludes archived and skipped annotations
    val singleAnnotations = query.count { it[Questions.annotationCount] == 1 }
    val desiredAnnotationCount = if (singleAnnotations > MAX_SINGLE) 1 else 0

    query
        // Select questions with one annotation if more than MAX_SINGLE
        .filter { it[Questions.annotationCount] == desiredAnnotationCount}
        .randomOrNull()
}
```

Figure D.2: Kotlin code for selection of the next question to annotate for a given user. Code is slightly altered for better readability.

E. QA Collection Form

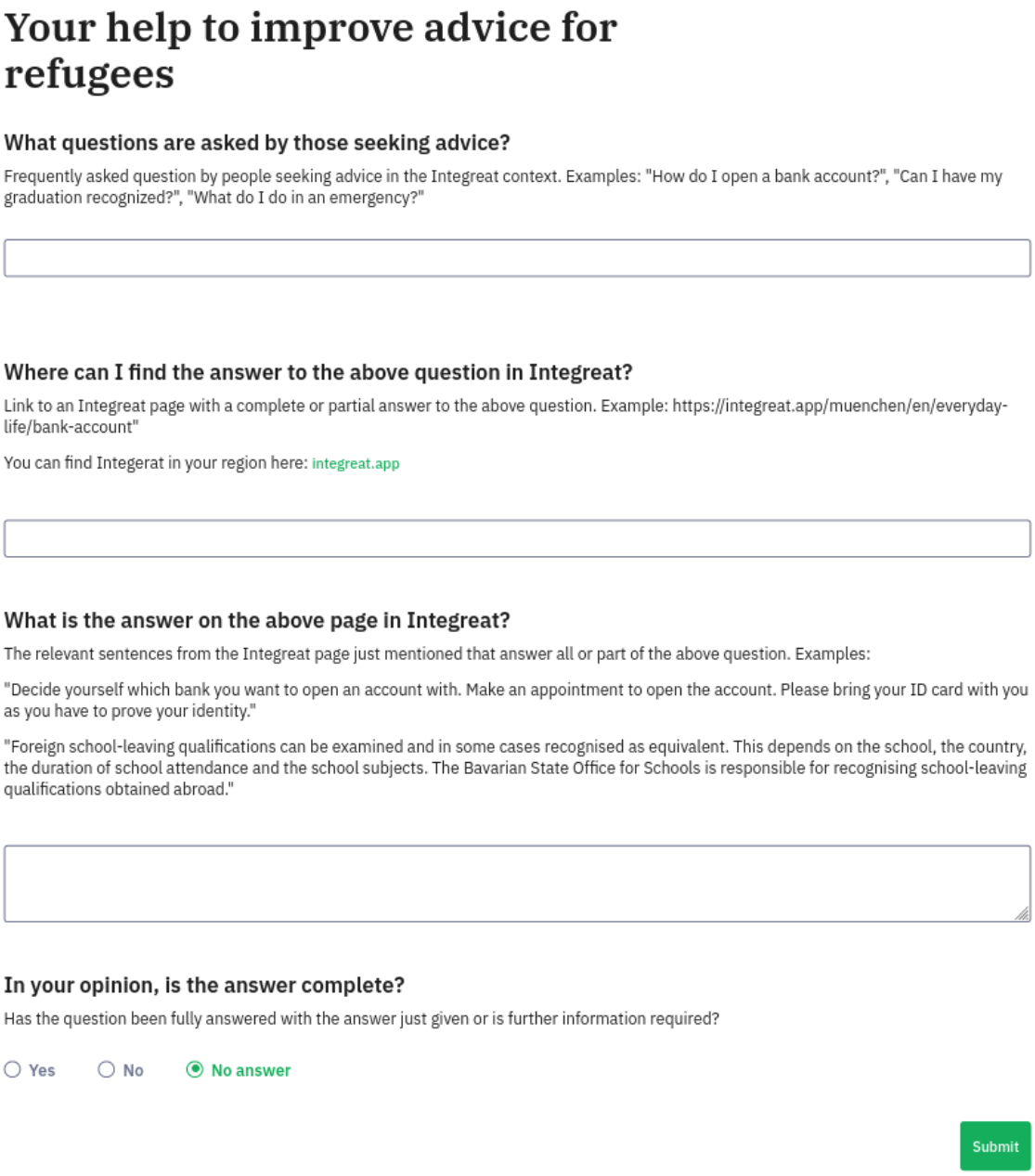

Figure E.1: Initial QA collection form. Selecting *No* as an answer to *In your opinion, is the answer complete?* shows additional input fields for more (partial) answers.

F. LLM Answer Sentence Indices Postprocessing

```python
import re


def extract_answer_lines(raw_input):
    try:
        answer_lines = []
        input = raw_input.replace('"', '')
        first_line = input.split('\n')[0]
        answers_start = input.index('[') + 1
        answers_end = input.index(']')

        matches_pattern = '[' in input and ']' in input
        # Extract text between brackets or first line otherwise
        raw_answers = first_line
        if matches_pattern:
            raw_answers = input[answers_start:answers_end]

        # Split answer parts
        answer_parts = [it.strip() for it in raw_answers.split(',')]

        for answer_part in answer_parts:
            if answer_part.isdigit():
                answer_lines.append(int(answer_part))
            elif re.match(r'[0-9]+\s*-\s*[0-9]+', answer_part):
                # Extend indices for ranges, e.g., '1-3' to '1,2,3'
                start, end = [it.strip() for it in answer_part.split('-')]
                if start.isdigit() and end.isdigit():
                    for index in range(int(start), int(end) + 1):
                        answer_lines.append(index)
        return answer_lines
    except Exception:
        return []
```

Figure F.1: Python code for extracting the answer sentences in the LLM text extraction setup.