

## Master Thesis

# Domain-specific adaptation of a large language model for the German administration

# Ehrenwörtliche Erklärung

Hiermit erkläre ich, David Wolpers, geboren am 21.03.1997 in Nürnberg,

- (1) dass ich meine Masterarbeit mit dem Titel:

**„Domain-specific adaptation of a large language model for the German administration“**

Im Betrieb SEITENBAU unter Anleitung von Professor Bittel und Dr. Heitmann selbständig und ohne fremde Hilfe angefertigt habe und keine anderen als die angeführten Hilfen benutzt habe;

- (2) dass ich die Übernahme wörtlicher Zitate, von Tabellen, Zeichnungen, Bildern und Programmen aus der Literatur oder anderen Quellen (Internet) sowie die Verwendung der Gedanken anderer Autoren an den entsprechenden Stellen innerhalb der Arbeit gekennzeichnet habe.

Ich bin mir bewusst, dass eine falsche Erklärung rechtliche Folgen haben wird.

Konstanz, 29.02.2024

David Wolpers

## Master Thesis

# Domain-specific adaptation of a large language model for the German administration

by

**David Wolpers**

in Partial Fulfilment of the Requirements for the Degree of

**Master of Science**

in Master Informatik

at the Hochschule Konstanz University of Applied Sciences,

Student Number: 295416

Date of Submission: 29th February 2024

Supervisor: **Prof. Dr. Oliver Bittel**

Second Examiner: **Dr. Benjamin Heitmann**

An electronic version of this thesis is available at <https://github.com/da721wol/Domain-specific-LLM>.

# Abstract

This paper evaluates the possibility of adapting a training process of a LLM to use domain-specific data during pretraining with the goal of training a cheap domain-specific LLM. Current approaches to domain-specific LLMs require labelled data, costly computational resources, or RAG, which has to be set up. Using domain-specific data during pretraining skips the requirement of labelled data. It also opens the possibility of increasing the capabilities of RAG or even skipping it altogether. By focusing on a domain-specific LLM with no generic capabilities, computational costs can be kept low as well.

To achieve this, the thesis trained two German T5 models with German administration data. One model was trained from scratch, while the other used a pretrained model from GermanT5 and added additional domain-specific pretraining. Different fine-tuning processes are conducted to show the effect of closed-book and open-book question answering. The models are evaluated and compared with existing models using RAG on common questions in the provided domain.

The results show that using an externally pretrained model is preferred due to the far lower costs while having similar or better scores across the board. Nevertheless, it didn't indicate a high likelihood that the approach of this thesis can result in high-quality domain-specific closed-book question answering. While larger models show slightly more promise, it is unlikely that a viable model can be trained. Open-book question answering shows more promise, even beating average-sized existing models.

Future research should therefore focus on answering open-book questions. Using larger models and better datasets might offer even better results while remaining viable when considering costs.

# Contents

<b>List of Abbreviations</b>	<b>vi</b>
<b>List of Figures</b>	<b>vii</b>
<b>List of Tables</b>	<b>viii</b>
<b>1 Introduction</b>	<b>1</b>
<b>2 Literature Background</b>	<b>4</b>
2.1 Transformer architecture . . . . .	4
2.1.1 Tokenization . . . . .	5
2.1.2 Embedding. . . . .	6
2.1.3 Encoder . . . . .	7
2.1.4 Decoder . . . . .	9
2.2 Training . . . . .	10
2.2.1 Parameters. . . . .	10
2.2.2 Hyperparameters . . . . .	11
2.2.3 Pretraining . . . . .	13
2.2.4 Fine-tuning. . . . .	14
2.3 Hardware, Platforms, and Frameworks. . . . .	15
2.3.1 Tensor Processing Unit (TPU) . . . . .	15
2.3.2 Google Cloud Platform (GCP) . . . . .	16
2.3.3 Hugging Face . . . . .	18
2.3.4 JAX and Flax . . . . .	18
2.4 Domain-specific Large Language Models . . . . .	19
2.4.1 Fine-tune a generic LLM. . . . .	19
2.4.2 Pretrain a generic LLM with domain-specific data . . . . .	20
2.4.3 Retrieval Augmented Generation. . . . .	21
<b>3 Methodology</b>	<b>23</b>
3.1 Training Data . . . . .	23
3.1.1 Pretraining datasets . . . . .	23
3.1.2 Administration Data. . . . .	24
3.1.3 Fine-tuning datasets . . . . .	25

3.2	Models . . . . .	26
3.2.1	T5 <sub>BASE</sub> - Pretrained model from scratch . . . . .	26
3.2.2	T5 <sub>LARGE</sub> - Use an existing German T5 model and add domain-specific pretraining . . . . .	27
3.2.3	Variants . . . . .	29
3.3	Hardware and GCP Setup . . . . .	29
3.3.1	TPU and CPU selection . . . . .	30
3.3.2	Virtual Machine . . . . .	31
3.3.3	Storage . . . . .	31
3.4	Training Process . . . . .	32
3.4.1	Tokenizer . . . . .	32
3.4.2	Pretraining . . . . .	33
3.4.3	Fine-tuning . . . . .	35
3.4.4	Hyperparameters . . . . .	35
<b>4</b>	<b>Results and Evaluation</b>	<b>38</b>
4.1	Training Results . . . . .	38
4.1.1	Training duration . . . . .	38
4.1.2	Cost . . . . .	39
4.1.3	Availability . . . . .	41
4.2	Evaluation results for common NLP benchmarks . . . . .	41
4.2.1	Open-Book Question and Answering . . . . .	42
4.2.2	Closed-Book Question and Answering . . . . .	42
4.2.3	Classification . . . . .	43
4.3	Evaluation of domain-specific questions . . . . .	44
4.3.1	Process . . . . .	44
4.3.2	Questions . . . . .	46
4.3.3	Candidate models . . . . .	46
4.3.4	Evaluation Results . . . . .	48
<b>5</b>	<b>Conclusion</b>	<b>50</b>
5.1	Using an available pretrained model is considerably cheaper than a model trained from scratch . . . . .	50
5.2	TPUs are cost-efficient compared to CPUs . . . . .	51
5.3	Closed-book QA doesn't show much promise . . . . .	51
5.4	Open-book QA might work . . . . .	53
5.5	Small domain-specific dataset . . . . .	54
5.6	Fine-tuning should focus on the required tasks . . . . .	54
5.7	Possible future adaptation of this work . . . . .	55

---

<b>References</b>	<b>56</b>
<b>A Question Answer Pairs</b>	<b>63</b>
A.1 Time or money-related question. . . . .	63
A.2 Location or source. . . . .	68
A.3 Yes/No-Questions . . . . .	72
A.4 Explanations . . . . .	75

# List of Abbreviations

<b>LLM</b>	Large Language Model
<b>AI</b>	Artificial Intelligence
<b>NLP</b>	Natural Language processing
<b>ULM</b>	Unigram Language Model
<b>BPE</b>	Byte Pair Encoding
<b>NN</b>	Neural Network
<b>FNN</b>	Feedforward Neural Network
<b>MLM</b>	Masked Language Modeling
<b>QA</b>	Question Answering
<b>RAG</b>	Retrival Augumented Generation
<b>DPR</b>	Dense Passage Retrieval
<b>CPU</b>	Central Processing Unit
<b>GPU</b>	Graphics Processing Unit
<b>TPU</b>	Tensor Processing Unit
<b>XLA</b>	Accelerated Linear Algebra
<b>GCP</b>	Google Cloud Platform
<b>ML</b>	Machine Learning
<b>VM</b>	Virtual Machine
<b>GKE</b>	Google Kubernetes Engine
<b>MXU</b>	Matrix Multiply Unit
<b>TRC</b>	TPU Research Cloud
<b>ROUGE</b>	Recall-Oriented Understudy for Gisting Evaluation
<b>SGD</b>	Stochastic gradient descent



# List of Figures

2.1	Transformer Architecture . . . . .	5
2.2	A fully connected multi-layer Feedforward Neural Network . . . . .	9

# List of Tables

2.1	Comparing TPUV2 and TPUV3 specs . . . . .	17
3.1	Configuration parameters for T5 <sub>BASE</sub> . . . . .	26
3.2	Datasets used to pretrain T5 <sub>BASE</sub> . . . . .	27
3.3	Configuration parameters for T5 <sub>LARGE</sub> . . . . .	27
3.4	Datasets used to pretrain T5 <sub>LARGE</sub> . . . . .	28
3.5	Datasets used to fine-tune all T5 variants . . . . .	30
3.6	Arguments for the tokenizer script . . . . .	33
3.7	Arguments for the pretraining script . . . . .	34
3.8	Arguments for fine-tuning scripts . . . . .	36
3.9	Hyperparameters for pretraining . . . . .	37
3.10	Hyperparameters for fine-tuning . . . . .	37
4.1	Training time . . . . .	39
4.2	Training costs . . . . .	40
4.3	Open-book QA - NLP Benchmarks . . . . .	42
4.4	Closed-book QA - NLP Benchmarks . . . . .	43
4.5	Classification - NLP Benchmarks . . . . .	44
4.6	Average Results for domain-specific Evaluation . . . . .	48

# 1

## Introduction

Large Language Model (LLM) are appearing more and more in everyday life. Even smaller use cases are trying to use the ever-increasing possibilities of these models, like searches in specific domains. Most searches nowadays are keyword-based. If you use Google, it just takes your query and returns matching documents and links for the keywords. The results often require additional steps from the user to obtain the actual answer. Asking a person the same question would result in a direct answer to the query, assuming that the person knows the answer. LLMs nowadays are much closer to a person answering a question than the old keyword-based search. Bing has already incorporated GPT-4 into their search engine because of this.

keyword-based search has another issue. The answer itself might not contain any word of the original question. This work uses the German administration, where some words can be very different from the common tongue. Everybody uses "car" for their everyday conversations, but official documents will always use "motor vehicle". A straightforward search for car keywords might therefore not return the desired result. While there are ways around this, there will always be exceptions where you don't find the thing you were looking for.

But to get a useful answer, the LLM needs to know what you're asking him about. Depending on the question, this is much easier or harder for an LLM to do. Questions that don't have a single correct answer are easier than ones that require one specific fact. And questions in a specific domain are even more complicated. If the LLM never had access to this domain, it would be impossible to answer the question correctly. This is no different from a person getting asked a question about a topic he has no knowledge about. For small company-specific domains, especially with sensitive data, this is very likely. This domain-specific Question Answering (QA) is the main focus of

this thesis.

Multiple approaches to this issue exist. The most common solution is to use a generic LLM to process the documents and look for the answer, most of the time as part of a Retrieval Augmented Generation (RAG) process. This is just reading comprehension, something modern state-of-the-art LLMs excel at. But this is also the issue with this approach. State-of-the-art LLMs are huge. Running these billions of parameter models anywhere is expensive, something most small use-cases just can't justify for the small advantage. And again, sensitive data might also prevent the use of external hosted models. Additionally, any new words these documents contain might be new to the model, resulting in less-than-ideal results. The second approach is to create a set of questions and answers and train the models with this set. This requires a lot of manual labour, and multiple relevant questions will probably be left out in the process.

Both of these approaches rely on a generic LLM to achieve domain-specific knowledge. Training these models with domain-specific data would be extremely costly; the largest models cost millions to train. To circumvent this, this thesis used a new approach to domain-specific question answering. For domain-specific cases, a generic LLM is actually not necessary, as domain-specific searches only want to answer domain-specific questions. They just need to understand the language and its connection to their knowledge of the domain. By removing this requirement, the LLMs can be drastically reduced in size. This is necessary to enable training from scratch using additional domain-specific data. In contrast to the second approach, the LLM gets access to all information in the domain without having to specify any answers.

The goal of this thesis is to show if it is possible to train new, small, and cost-efficient models for domain-specific QA. To achieve this goal, existing training scripts are adapted to include the available German administration data during the pretraining process. The chosen training scripts are then used to train a German T5 model for the German administration. Its only goal is to answer questions concerning anything in the German administration. It can't tell you the last World Cup winner in football, but it might be able to tell you the cost of a new passport.

Costs are one important factor in ensuring viability, even for small companies and domains. To incorporate this factor even further, a second, cheaper alternative is explored. Instead of training the model completely from scratch, an existing small German T5 model has an additional purely domain-specific training step added. This approach uses the same training scripts as the first approach.

For both of these models, this work highlights the associated costs of training. It also explores different fine-tuning approaches to see if any result in better answers. All variants will then be compared to external LLM using RAG to show how they compare to the previous approach to domain-specific QA.

Training a model is an expensive procedure, even for the small models used in this thesis. To reduce costs, the LLMs used in this work are smaller than they probably should be. The resulting LLMs are therefore too small for practical applications but large enough to show differences between the different approaches.

The next chapter covers the literature concerning LLM and how they work. It also gives more details about the current approaches to domain-specific QA. Finally, it describes the necessary hardware and platforms for training an LLM model.

The third chapter focuses on the training process used for this thesis. It describes the exact data used for training and the different models and their variants trained with this training data. It also explains any hardware and platform choices. Lastly, it describes the exact training process, including used scripts and arguments.

The training results are described in Chapter 4. This includes an overview of the costs and duration. Additionally, common Natural Language processing (NLP) benchmarks give insight into how the models understand the German language. Finally, the evaluation process for domain-specific questions is described, and the results are presented. The last chapter combines the found observations to answer the question of whether it is possible to train a domain-specific LLM with the approach used in this thesis.

# 2

## Literature Background

Over time, multiple different LLM architectures have been proposed. This literature overview will focus on the Transformer architecture, which is the most common architecture nowadays [Vas+17]. It will also show how a LLM with a transformer architecture can be trained. How these concepts are implemented in the T5-Model ([Raf+19]) will also be highlighted. The last section will describe current approaches to domain-specific LLMs and what prevents them from being the perfect solution to this problem. Large Language Models are Artificial Intelligence (AI) models that focus on NLP. The general goal of NLP is understanding human languages. LLMs will always take some form of text input to produce some text as an answer. This is the specialist form of the more general generative AI. Generative AI takes any media input, like images or audio, and generates some form of media as a result [Kur23]. This can be text, but for example, AI-models like DALL-E produce an image that fits the provided prompt [Ope21]. The "Large" part in the name stems from the large amount of data necessary to train these models.

### 2.1. Transformer architecture

Most LLMs nowadays are based on a transformer architecture [Vas+17]. This includes the T5 model used in this work [Raf+19]. Like most successful LLMs, it is split into an encoder and a decoder. The goal of the encoder is to extract the features of the input. These features are then passed to the decoder to generate the output. The output gets generated one token at a time, where each generated token is then passed as an

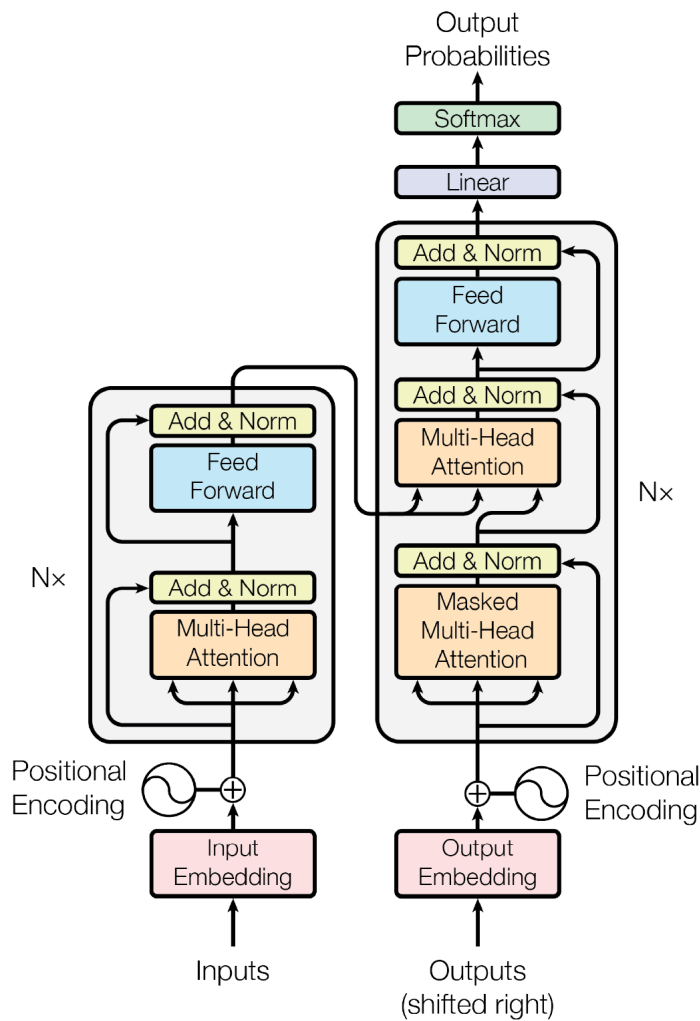


Figure 2.1: Transformer Architecture  
Quelle: [Vas+17]

additional input to the decoder. The output will therefore always depend on previously generated tokens in an auto-regressive manner [Gra13]. This process will repeat until it reaches a predefined maximum or a special end-of-sentence marker  $\langle EOS \rangle$ . [Vas+17]

### 2.1.1. Tokenization

The initial input for the LLM needs to be processed before being passed to the encoder. The first step is always a tokenization step. A LLM doesn't process a sentence word by word. Instead, it splits the sentence into individual tokens. Short or common words can be a single token; e.g., "the" would be a common example of this. Other words are

split into multiple tokens. One possible example of this are verbs in various conjugated forms. Common verb stems stay as one token, while common endings like "ing" are a different token. Tokens also include punctuation or special characters. These different tokens form the vocabulary of the LLM. The vocabulary should always be a set of tokens that can represent the dataset in the best possible way. The best possible way can be defined slightly differently, depending on the tokenizer. But the general concept is always trying to use the least number of tokens for any given word, where words with higher frequency are also weighted higher. [Hum23]

The size of the vocabulary is defined beforehand. Common sizes are in the mid-tens of thousands. The exact vocabulary depends on the dataset, language, and tokenization algorithm. While the original paper for T5 uses a SentencePiece WordPiece Tokenizer, this work uses a slightly customised version of a SentencePiece Unigram Tokenizer [KR18]. Unigram tokenization was first described in [Kud18] and is based on Unigram Language Model (ULM). In language modelling, the goal is to predict the next word by assigning a probability to each word. ULM assumes that the probability of each word occurring is independent of any previous words. The probability of the next word is therefore only determined by how often the word occurs in the given dataset.

Like most tokenization algorithms, it requires training to learn the vocabulary for the given dataset. In contrast to other common algorithms like Byte Pair Encoding (BPE) ([Gag94]) or WordPiece (first mentioned in [SN12]), a very large initial vocabulary will be created. SentencePiece uses an efficient algorithm based on the Enhanced Suffix Array to create the initial vocabulary [AKO04]. Using this initial vocabulary, the algorithm calculates a loss over the given dataset. Afterwards, it takes any non-basic token, meaning any token with a length larger than one, and calculates the loss without that token. A predefined percentage of tokens with the lowest overall additional loss is then removed. This process is repeated until the desired vocabulary size is reached. [Kud18]

### 2.1.2. Embedding

The transformer model works with vectors and matrices of scalar values at every step. Since a token has no scalar value, the next step is converting all tokens into their vector representation. The spelling of a token doesn't contain any information. Therefore, the only direct goal of this representation is to differentiate between the tokens. One very easy approach to this would be representing the tokens in a one-hot encoding [HH12]. One-hot encoding assigns each token a vector of size  $n$ , where  $n$  is the size of the vocabulary. Every entry in the vector is 0, except one with a 1. The issue here is that this results in a huge vector for each token. If you combine all these vectors into one input matrix, the result will be a matrix of  $n \times n$ , where  $n$  is the size of the vocabulary.



[Gan23]

To circumvent this problem and gain some new desirable properties, an embedding layer can be used. The embedding layer consists of a single embedding matrix that converts the one-hot vectors to smaller embedded vectors. The number of columns  $n$  is the vocabulary size. The number of rows  $m$  is the lower-dimensional vector size (the embedded vector size). If we take the product of this matrix with a one-hot encoded vector, the result will just be a column in the matrix. This approach has two advantages. The key advantage is the fact that the resulting vectors are much smaller. The other advantage comes from the fact that this embedding matrix doesn't have to be fixed. Instead, it can change its values during training. The result of the training would be that similar tokens, like synonyms, also have similar embedded vector representations.

[Vas+17]

In the last step, these embedded vectors are then enriched with positional information. [Vas+17] uses predefined and not learned positional embeddings, as in [Geh+17]. This information includes the position of a token in a given input sentence. Without this information, sentences like "Simon loves the dog" and "The dog loves Simon" are the exact same [Kik21a]. This is necessary because the transformer architecture doesn't use any form of recurrence or convolution, which was used in previous architectures to gain positional information. [Vas+17]

### 2.1.3. Encoder

These embedding vectors are then passed to the encoder. The encoder consists of multiple identical transformer blocks. Each block is split into two parts: a multi-head attention layer and a Feedforward Neural Network (FNN). Attention is the process of looking at individual tokens in the context of every token in a sentence. Most commonly, this is done between the input tokens and the already-generated output tokens [Geh+17]. The main reasoning behind attention is to focus on the important aspects of all previous tokens, even if their number is large. To calculate attention, the attention function is defined, which maps a query and a set of key-value pairs to an output. Query, keys, and values are vectors created by multiplying the previously mentioned embedding vectors (which are also the result of every layer in a transformer architecture) with different matrices. Each head in an attention layer has a different trainable matrix for the query, the keys, and the values. Using these vectors, the computed output is a weighted sum of the values. Each value's weight is determined by a compatibility function that compares the query with its corresponding key. [Vas+17]

### Self-attention

Attention was usually calculated between different sequences. The query was therefore from one sequence, and the key-value pairs were from a different sequence. The transformer architecture introduces self-attention, where the sequence is compared against itself. Query, keys, and values all come from the same sequence. Previous architectures used recurrent and convolutional layers to gain contextual information about the input. In contrast, the encoder block relies on self-attention for the output of the previous encoder layer. For the first block, this is the embedded input sequence. [Vas+17] The contextual information is important for different aspects, like differentiating between two words with the exact same spelling, like in "I see the bank of a river" and "I pick up money from the bank". Just like general attention, it also helps to focus on important aspects of the sentence, since tokens that don't have any big correlation to any other token are probably not that important. [Kik21b]

Since each input sequence has different forms of context throughout the sequence, it was shown that it is advantageous to use multiple smaller attention heads instead of a single large head. Each head has its own smaller matrix, where  $h$  different heads split the single head dimension  $d$  matrices into  $h$  different  $d/h$ -dimensional matrices. During the training process, each head can learn to focus on different forms of context independently. Due to averaging, this is not possible with a single attention head. Using multiple larger heads with dimension  $d$  would increase the computational complexity without gaining any significant qualities. The results of each attention head are then combined to again achieve a vector of dimension  $d$ , which is added to the original embedding vectors and normalised. This prevents the mean and standard deviation of embedding vector elements from moving around, which makes training unstable and slow. [Vas+17]

Self-attention is the defining factor that differentiates the transformer architecture from other approaches. Due to its nature, it is possible to calculate the complete multi-head attention in parallel by combining all vectors in matrices. The older recurrent architecture approach had to calculate  $n$  sequential operations, where  $n$  is the sequence length. [Hoc+01] also showed that it is easier to learn long-range dependencies the shorter the path is. Since every token in an attention layer is connected to every other token, this shortest path is a constant one for self-attention. [Vas+17]

### Position-wise Feedforward Neural Network

The second step of a transformer block is a position-wise FNN. In a Neural Network (NN), neurons are connected to each other between multiple layers. Each neuron has weighted input from multiple neurons and creates a single output via an activation function. The exception to this is the input layer, which just takes the values passed to the NN as an output. If no output affects any previous inputs, the network is a FNN. Ad-

ditionally, the network is fully connected when the output of every neuron is connected to every neuron in the subsequent layer. If there are more than two basic layers (the input and the output layer), the network is called a multi-layer network, as depicted in 2.2. The extra layers are called hidden layers. [Saz06]

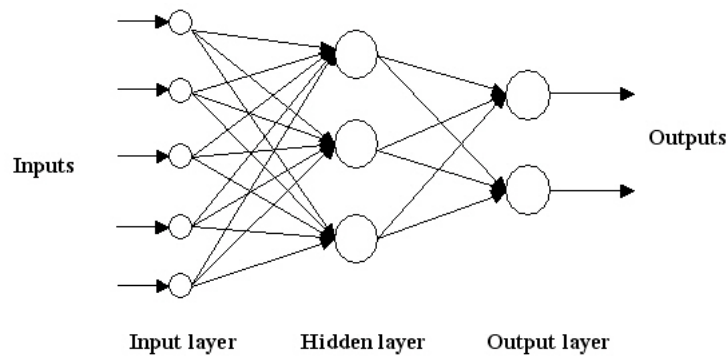


Figure 2.2: A fully connected multi-layer Feedforward Neural Network  
Quelle: [Saz06]

The FNN used in [Vas+17] is a fully connected multi-layer network, which consists of just one hidden layer. The hidden layer used is about four times the size of the input and output layers. T5 adapts this approach slightly by using a second hidden layer with identical dimensionality [Raf+19]. It is "position-wise" because each embedding vector goes through the same network independently. The input and output sizes are therefore just the size of the embedding vectors. The results of these FNN are then again added to the embedded vectors and normalised.

Multiple of these transformer blocks ([Vas+17] uses six blocks for both the encoder and the decoder) are then stacked to form the encoder. The exact number of blocks depends on the configuration used; for T5, this can be anywhere between 6 and 24 encoder and decoder blocks. The number of encoder and decoder blocks should always be the same. [Vas+17]

#### 2.1.4. Decoder

For the decoder, this process is very similar, but the transformer block has one additional layer. The first layer uses the embedding vectors of the previous layers, just like the encoder. This includes the output and therefore potential tokens, which haven't even been generated. Since the number of embedding vectors is always the same (any missing tokens are just filled with a special  $\langle PAD \rangle$  token or are already defined during a training process), this process should be careful to only rely on positions less than  $i$  for predictions of  $i$ . Via masking and shifting by one to the right, this can be

ensured. The result is again added to the embedding vectors and normalised.

The next layer is another attention layer, but this time keys and values come from the embedding vectors of the encoder and the query from the embedding vector of the previous layer. This is therefore not a case of self-attention. The final layers are copies of the encoder layers. And just like the encoder, the decoder also stacks six of these blocks. [Vas+17]

Finally, a learned linear transformation like in [FJK96] is applied to the resulting vectors of the final block. This results in a single vector of scalar values for each possible token. Using this vector in a *softmax* function as described in [BN06] converts these values to probabilities. With a greedy approach, the generated token would be the token with the highest possibility. [Vas+17] opted to use a beam search with size four instead. Instead of just choosing the next token, all possible sequences of  $n$  tokens will be analysed. The next token will then be the first token in the most likely sequence of  $n$  tokens. [Wu+16] A different variant of the transformer architecture is a decoder-only network. As the name suggests, it skips the encoder blocks, using only decoder blocks. The most common examples are GPT models [Rad+18].

## 2.2. Training

Training a LLM is the process of tuning the parameters in the model. In a transformer architecture, multiple different parts contain trainable parameters. The embedding layer learns to group similar words; the attention layer learns relationships between different pieces; and the FNN learns to use this information to "understand" the sentence. This is often split into an unsupervised and a supervised step. For pretraining, an unsupervised process is used, while fine-tuning uses supervised processes. [DeI21]

### 2.2.1. Parameters

Modern LLMs are often defined by their size, where the size is the number of trainable parameters they have. A large part of the recent improvements in the field of NLP with LLMs can be credited to the ever-increasing number of parameters. Nevertheless, while more parameters can lead to more abilities and higher-quality outputs, they don't have to. To actually exploit the advantages of more parameters, larger datasets and extensive training durations are required. The key issue is that these training setups also get very expensive. This is particularly true because parameters in early layers are trained slower than parameters in later layers due to the nature of Stochastic gra-

dient descent (SGD) and Back-Propagation (see 2.2.3 for more detail) [Ama93]. More parameters are often a result of additional layers, which increases the issue. State-of-the-art open-source models like LLama work with up to 70 billion parameters. The training costs for these kinds of models can exceed millions. [Tou+23] For research purposes, much smaller and more manageable LLMs are often used. A common range is somewhere between 100 million and 300 million; this is equivalent to T5s "small" or "base" configurations [Raf+19]. While this decreases training costs and duration, dedicated training hardware is still necessary to get decent results.

### 2.2.2. Hyperparameters

In contrast to parameters, hyperparameters are non-trainable parameters that have to be selected by hand. Hyperparameters control the learning process in multiple ways. Depending on the model, use case, and training data, different hyperparameters can lead to different results. The two most common issues that can be avoided with the correct hyperparameters are overfitting and underfitting. Overfitting results in parameters that are too close to the training data after training, therefore losing their generic properties. Underfitting is the opposite, where the parameters are too far away from the training data after training, thus never gaining the ability to find more nuanced aspects. For this thesis, overfitting is actually desirable to some extent, since losing generic properties while gaining domain-specific knowledge is preferable. Multiple different approaches for finding as close to optimal hyperparameters as possible have been described, for example:

- Grid Search: List every reasonable value for each available hyperparameter. Then test each combination of values against a small subset of the training data. Finally, select the combination with the best result. [HCL+03]
- Random search: Instead of testing every combination, multiple random enumerations of hyperparameter values are selected for testing. This is much faster when the number of configurable hyperparameters increases. [BB12]
- Population-based: Multiple training processes are run simultaneously with different hyperparameters. Models that perform worse are slowly replaced with new models that use modified hyperparameters of better-performing models. This process avoids using one single set of hyperparameters for the whole training and instead adapts them on the fly. [Li+19]

All these approaches are lengthy and costly. This becomes especially noticeable for small models, where the actual training time is considerably smaller than for large mod-

els. But even larger models like BERT don't put a huge focus on hyperparameter optimisation; commonly used values should therefore suffice for most use-cases [Dev+18]. Common hyperparameters include:

- Training steps: the number of training steps.
- Batch size: the data is often provided in batches and not in single instances; this parameter defines the size of these batches.
- Sequence length: the number of tokens the model can receive at once.
- Learning rate: the learning rate defines how much the parameters should change during each step.
- Weight decay: discourages large weights by penalising them.

Training steps, batch size, and sequence length combine to describe the amount of training the model has gone through. The more training steps, the larger the batch size, and the longer the sequence, the better. Together, the number of tokens the models see during training is higher, which should lead to better training results. The disadvantages of increasing these parameters are a longer training duration for more training steps and an increase in the required hardware memory and training duration for a larger batch size or sequence length. Nevertheless, larger batch sizes also come with the advantage that concurrent calculation possibilities of the underlying hardware (see 2.3 for more detail) can be utilised further. And a longer sequence length allows the model to accept and understand longer input after training. There is a limit to improved training with these parameters. This depends on multiple factors, but the key aspects are the size of the model and the size of the datasets used. If the used data is too small or similar, overfitting may occur after some time, and if the model is too small, it may reach its peak before the training is finished. Therefore, larger datasets and models enable the possibility of a longer training duration.

The adaptive approach from population-based or similar processes can be used in an easier form. A common approach for the learning rate, for example, is to start with a higher rate and slowly decrease it the longer the training is proceeding. You can also avoid providing a number of training steps and instead use a process where you stop as soon as your loss function reaches a desired level. Sometimes the amount of available data is also limited, restricting the maximum number of steps without using the same data multiple times. This can be achieved by using multiple epochs during training.

### 2.2.3. Pretraining

Pretraining is always the first step in training a new transformer-like LLM. The main goal of pretraining is to gain general knowledge about language as well as specific facts. Due to the huge amounts of data used for this step, this is by far the most time-consuming part of the training process. Large state-of-the-art models like LLama use multiple terabytes of data for training [Tou+23]. This is possible because pretraining is an unsupervised process, which means no data has to be labelled beforehand. Labelling is the process of providing an expected output to a specific input [Del21]. Instead, in unsupervised training, the expected output is automatically generated from the input. While there are different ways of achieving this, nowadays this is often done by some form of Masked Language Modeling (MLM). A common example is BERT [Dev+18], which randomly masks 15% of an input sentence with a different token. The new token can either be a special  $\langle MASK \rangle$  token or a random, different token from the vocabulary. The goal of the LLM is then to predict the correct tokens. T5 slightly altered this approach while also including some parts of the “word dropout” regularisation technique from [Bow+15]. [Raf+19] also masks 15% of the tokens. In contrast to BERT, these replaced tokens are assigned a unique id to differentiate between the different masks. Additionally, consecutive spans of replaced tokens are only replaced by one unique token. A T5 model then needs to predict the actual tokens.

In both cases, the parameters are then adjusted using an optimizer. The original paper [Vas+17] used an Adam optimizer ([KB14]), while T5 switched to AdaFactor, a memory-efficient approach to Adam [SS18]. [Raf+19] An optimizer updates the parameter values based on SGD and the learning rate. SGD is the process of finding the minimum of a (loss) function by using the negative gradients of all parameters [Ama93]. The loss function used for this is usually a cross-entropy loss function [BN06].

To calculate these gradients for all parameters, back-propagation is used. This process has one downside that becomes particularly noticeable in large multi-layer networks. Back-propagation starts calculating the gradients at the last layer, since this is also the layer used to calculate the loss [Ama93]. It then proceeds through the network layer by layer, calculating the gradients of all parameters. [Nie15] shows that this process leads to smaller gradients the further in the front a layer is. Since gradient size affects the learning speed, this vanishing gradient problem results in slow learning for early layers compared to later layers. Having more layers worsens this effect further. This increases the required training time for large models exponentially compared to the linear increase a larger parameter count might suggest. GELECTRA, a German model, has a 15-times longer total training duration for a model while increasing the parameter count only threefold [CSM20].

### 2.2.4. Fine-tuning

After pretraining, the resulting model should have a general "understanding" of the languages it has been trained in. But most of the time, this is not enough to handle specific tasks. Fine-tuning wants to address this issue. A model can be fine-tuned for multiple different tasks. Common tasks include:

- Open-book QA: Answer a question with some context that contains the answer somewhere.
- Closed-book QA: Answer a question without any additional context.
- Classification: Which label fits best with the provided input text? The possible labels are predefined; a common example would be *bad*, *good*, or *neutral*.
- Summarising: Summarise the provided text in very few sentences.
- Question Generation: Generate the question to a provided answer.

In contrast to pretraining, fine-tuning is a supervised process; any data used must therefore be labelled. For these tasks, the label is always the expected answer. During training, the model is then provided with the input, and the loss is then calculated compared to the labelled answer. While fine-tuning helps to handle specific tasks, it has been shown that LLMs can also deal with new tasks they've never been fine-tuned on [TS10]. For the small models used in this thesis, this behaviour is not expected to be noticeable.

### T5

For T5, every language problem needs to be converted into a text-to-text format. By doing this, every task can also be trained in the same way, making it easier to fine-tune different tasks. Open-book QA can be used as an example, which doesn't use two inputs but instead one input, including both the question and the answer. Since all tasks have exactly the same form, it's harder for the model to differentiate between tasks. To circumvent this issue, T5 includes a task-specific prefix with the input. For question generation, this could be *What was the question?:* for example. The goal of this approach is to increase any transfer learning possibilities. [Raf+19]



## 2.3. Hardware, Platforms, and Frameworks

Training an LLM requires extensive amounts of operations, so many in fact that training in a reasonable time becomes impossible on a Central Processing Unit (CPU). As previously mentioned, all values of a model are stored in different matrices, and any interaction with the model, be it training or inference, is just matrix calculations. Efficient matrix calculation splits the calculation into smaller and smaller sub-matrices, down to scalar values. This process is predestined for parallel execution on multiple cores. CPUs main advantage is flexibility, which makes it possible to run any application or calculation on a CPU. While parallel operations are possible on CPUs, even the current record of 288 cores only does so to a limited extent [Ken23].

In contrast, even cheap and old consumer-grade Graphics Processing Unit (GPU)s can easily exceed this number by an order of magnitude. GPUs can gain this increased efficiency by specialising in parallel calculations, while losing the flexibility of CPUs. Most, if not all, LLM training is conducted with some form of GPUs. But even GPUs still need to handle different forms of calculations or applications. And both CPUs and GPUs require access to slow shared memory, reducing the possible speed even further.

### 2.3.1. Tensor Processing Unit (TPU)

In 2016, Google presented an alternative architecture with Tensor Processing Unit (TPU)s. TPUs lose all flexibility with the sole purpose of efficient parallel tensor calculations [Jou21]. TPUs excel at AI training or inference, since both of these use cases mainly consist of tensor calculations. In contrast, any other use case isn't suited for TPUs at all, in contrast to GPUs and especially CPUs. A tensor is an any-dimensional arrangement of scalar values. So both a vector and a matrix are tensors of different dimensions. The whole architecture of a TPU is designed to do matrix calculations in the most efficient way. This comes down to the physical implementation, where sub-results from matrix calculation can be passed on to the next calculation unit, which is also right next to it on the chip. The downside to this approach is that the data and model used to train with a TPU must have certain shapes for maximum efficiency (or to run at all). [Goo24b]

To even run any training on a TPU, the code has to be compiled by the XLA compiler, as no other option exists. Any Machine Learning (ML)-framework emits a graph containing the linear algebra, loss, and gradient of a model for training or inference. Accelerated Linear Algebra (XLA) then compiles these components into TPU machine code, which can then be run on a TPU. But XLA isn't restricted to TPUs; multiple other platforms

are also supported for their high-performance ML compilation. [Goo24b]

The architecture of a TPU consists of a 128x128 systolic array to calculate the results and subresults for each operation. To gain maximum efficiency, the data batches passed to the TPU should have a shape that can easily fit into these dimensions. Additionally, the TPU memory subsystem prefers feature dimensions that are multiples of eight. If both of these requirements are met, XLA can use them for maximum tiling efficiency.

To increase performance even further, TPUs support the BFLOAT16 floating point format. Different floating-point forms offer varying amounts of precision. While ML doesn't require the highest amount of precision, like the 80-bit FP80 format, some precision is necessary. Most LLM training is conducted in the FP32 format, while FP16 is used for faster but less precise training. Another common approach is decreasing the floating point format for inference to increase the output speed. Google developed BFLOAT16 for ML, ignoring bits that didn't help ML processes while focusing on the others. The result is a shorter format, which enables higher performance while producing similar results to FP32. [Kal+19] TPUs only support FP32 and BFLOAT16 but not FP16.

TPUs have been released in five different versions over the years, with an ever-increasing speed and memory size. This thesis uses versions 2 and 3 for training. Both versions consist of four chips with two tensorcores each. TPUv3 has twice the number of Matrix Multiply Unit (MXU)s per tensorcore, the main component for the actual matrix calculations. Each MXU contains the previously mentioned 128x128 systolic array and its own memory. Due to this, both the available memory and performance increased twofold compared to the previous iteration. Additionally, the clock and memory speeds are also slightly higher, which results in a proposed speedup of about 2.7x, ignoring any memory limitations. An overview can be seen in the table 2.1. [Jou+20]

It is possible to combine multiple of these devices into pods to increase the number of usable computing power.

### 2.3.2. Google Cloud Platform (GCP)

Training a model, especially a domain-specific model, is a one-time process in most cases. It is therefore not cost-efficient to buy the required hardware. Instead, cloud services, which provide the hardware for an hourly fee, are used as an alternative. This is especially true for TPUs, since TPUs can't be bought. The only available access is over the Google Cloud Platform (GCP), a cloud provider for multiple use cases, from storage over hosting a website to hardware renting. TPUs can be accessed in multiple ways. Vertex AI provides a framework tailored for ML but lacks the flexibility of using a custom training setup. Cloud computing provides access to TPUs with a Virtual

	TPUv2	TPUv3
Chips/Device	4	4
Peak TeraFLOPS/Chip	46 (16b)	123 (16b)
Clock Rate (MHz)	700	940
Memory size (on-/off-chip)	32MiB/16GiB	32MiB/32GiB
Memory GBs/Chip	700	900
Cores/Chip	2	2
MXUs/Core	1	2

Table 2.1: Comparing TPUv2 and TPUv3 specs  
Quelle: [Jou+20]

Machine (VM) for more flexibility during the training process. And GKE (Google Kubernetes Engine) goes even further, providing flexible access with the possibility of flexible scaling to multiple TPUs. [Goo24a]

GCP host their resources in different regions. Each region has slightly different costs and available hardware associated with it. Additionally, each region consists of multiple zones. Zones are mainly to separate instances for increased availability in case of a possible downtime in one zone. Other zones restrict their access to certain user groups. This is true for "us-central1-f", which can only be accessed by TPU Research Cloud (TRC) users. [Goo24c]

The TRC is a programme from Google that provides researchers with free access to TPUs for 30 days. The free TPUs are limited to certain versions and zones, including exclusive access to "us-central1-f". This work has access to the TRC and could therefore use some TPUs for free. Any TPU-associated costs in this thesis are therefore estimated based on available prices and training duration. [Goo24d]

TPUs in GCP are named according to the number of cores available. The smallest available TPU is a single TPU device, which according to the table 2.1 has eight cores for v2 and v3 devices. Consequently, their names are TPUv2-8 and TPUv3-8, respectively. The smallest available pod combines four devices for a total of 32 cores and is named TPUv2-32. V3 TPUs are named accordingly. Larger pods come in steps of power two, but some steps might be skipped, depending on the region and version. The largest v2 pods have a total of 512 cores, while the largest v3 pods go as far as 2048

cores.

### 2.3.3. Hugging Face

Training a model requires multiple prerequisites. A model configuration defining the exact structure of the architecture, like the number of decoder and encoder blocks, has to be created. Vast amounts of varying data have to be available and prepared. And to start the actual training, both of these aspects have to be combined with training scripts. While all these elements are available to themselves, Hugging Face provides a platform that gives access to all these elements at once. Hugging Face is tailored towards open-source AI development. To encourage this, the Hugging Faces platform provides a place for developers to publish their open-source models and datasets. Anyone can then use these for further development or practical applications. Available models include the largest open-source models, like LLama2. To ease the use of these models for further development, the platform also contains extensive documentation for different models, datasets, and training setups. [Jai22]

The main goal of Hugging Face is to encourage more AI development. To enable this even further, they released their Python Transformer library. Hugging Face describes it as "APIs and tools to easily download and train state-of-the-art pretrained models". While this is true, Transformers also provides APIs and scripts to enable the training of a new model from scratch. The Transformers library easily integrates common ML frameworks like PyTorch or TensorFlow. These frameworks provide the actual training code; Transformers just provides easy-to-use APIs and enables easy switching from one framework to the other. Additionally, the library provides full access to the Hugging Face platform, enabling the use of any available datasets or models for training. Most common models have a detailed description of how to train, fine-tune, and use them with the Transformer library. Each of these models also has available model configuration and tokenizer classes. [Wol+19]

### 2.3.4. JAX and Flax

As mentioned in Section 2.3.1, XLA is required to train a model on a TPU. While common frameworks like TensorFlow or PyTorch support XLA, JAX is the only framework that relies entirely on XLA for compilation. JAX combines the power of XLA and Autograd to achieve "high-performance numerical computing", gaining even better performance than XLA on its own [JAX24]. JAXs approach uses "composable function transformation" to achieve "more maintainable, more scalable, and more performant"

code, setting it apart from TensorFlow or PyTorch [Fla23]. "Autograd can automatically differentiate [between] native Python and Numpy code" [DT23]. Autograd's main focus is gradient-based optimisation, which is necessary for SGD and is therefore one of the most common calculations during training. Since Autograd was completely incorporated into JAX after the development of JAX, any future active development has only been done through JAX. [DT23]

The Flax framework wants to exploit these advantages even further while eliminating their weaknesses. Flax enables more flexible access to the capabilities of JAX while being easier to use. Flax provides easy access to training examples that can be forked and modified for any use case. With the linen model, it is possible to define and train a completely new model. Another advantage of Flax is its existing integration into the Hugging Faces Transformer library. While the same is true for JAX, Flax is integrated seamlessly, including examples and classes for a T5 model. According to Hugging Face, training on a TPUv3-8 with Flax is about 50% faster than with XLA and Pytorch and nearly three times as fast as Pytorch training on 8 V100 GPUs, which have a peak performance of 112 TeraFLOPS each. [Fac24c]

## 2.4. Domain-specific Large Language Models

To evaluate the quality and viability of the domain-specific approach in this thesis, it has to be compared against the current existing approaches. Current approaches to domain-specific LLMs can be grouped into three types. An existing generic LLM can be fine-tuned with the necessary data; a generic LLM can be pretrained from scratch with additional domain-specific data; or the matching context can be passed to a generic LLM during inference (the process of actually using the LLM). All approaches have in common that the LLM has to be relatively or fully generic LLM to handle the tasks.

### 2.4.1. Fine-tune a generic LLM

Producing a lot of labelled data focused on a specific domain for fine-tuning is a common approach to gain domain-specific knowledge. One example would be MedPaLM, which was released recently in 2023 by Google Research. It's a fine-tuned PaLM model (with 540 billion parameters [Cho+23]) with additional medical data. Therefore, it excels at answering medical questions. To increase the quality of answers, further instruction-prompt-tuning has been used.

The main issue with this approach is the required data. The data for MedPaLM is a com-

bination of existing medical QA datasets and common Google searches for medical-style questions [Sin+23]. Most domains have neither the luxury of preexisting labelled data nor Google's extensive data when it comes to domain-specific Google searches. This is especially true for generic questions in a domain-specific setting. Since answers to questions aren't easy one-to-one solutions, creating this data is mainly manual work, which is a very time-consuming process. There are approaches to creating labelled data for fine-tuning using existing state-of-the-art LLMs, as the Vicuna LLM has shown [Chi+23]. This approach should also be applicable to generate answers for domain-specific questions in a semi-automated way by combining the approach with RAG (Section 2.4.3 describes this process RAG in more detail).

Nevertheless, this approach requires the manual creation of questions if no existing search data exists. So one way or another, manual labour is required. And another issue comes from this fact. The resulting LLM doesn't actually learn the information of the domain; it just learns answers to the predefined questions. The quality and number of the questions therefore strongly determine the quality of the LLM for the specific domain. The manual work required for large domains could easily outweigh the advantages of using an LLM compared to a generic key-based search. The advantage of this approach is that the model can be smaller if the focus is on a smaller domain or less generic properties.

#### 2.4.2. Pretrain a generic LLM with domain-specific data

The approach closest to this paper would be pretraining a generic LLM with domain-specific data. An example of the second approach is BloombergGPT. Bloomberg is a huge financial company, and BloombergGPT represents this approach by focusing on financial questions. As the name suggests, BloombergGPT is a GPT model [Rad+18]. This GPT model was trained from scratch using the huge amount of financial data Bloomberg has access to. The key difference compared to the models trained for this thesis is the fact that the model is still generic. While the focus is on financial questions, the model can still exceed some open source models at generic, common NLP benchmarks. According to Bloomberg, the model has about 50 billion parameters, a number in the range of state-of-the-art open-source models like LLaMa. [Wu+23] This approach requires the same kind of hardware as any large generic model, while also requiring vast amounts of domain-specific data.

Both of these requirements make it unfeasible for small companies to use this approach. It would be far too expensive, and most domains just aren't large enough to create the necessary amount of data.

### 2.4.3. Retrieval Augmented Generation

The first two approaches both require vast amounts of (manually created) data, and in the case of BloombergGPT, extensive computing resources. Due to this, the most common approach nowadays is RAG (Retrieval Augmented Generation). RAG was first described in the paper [Lew+20] as an AI framework for improving the quality of LLM-generated responses by adding externally available context for the prompt. Using domain-specific documents for this can enable the LLM to answer domain-specific questions. This process changes the type of task for the LLM from a very difficult closed-book QA to a pretty straight-forward open-book QA.

And even outside of this approach, implementing RAG in an LLM-based QA system has another additional benefit. It ensures that the model has access to the most current, reliable facts. The training data used to train the parameters of an LLM has to end on some date. Any recent event happening after this date cannot therefore be stored in the parameters without additional training. But the documents provided with RAG can be updated independently of the LLM and therefore always contain up-to-date facts. Using the internet can also be seen as a form of RAG providing access to any publicly available domain.

Another advantage is the increased trustworthiness of a LLM. Validating the answer of a LLM is often difficult or even impossible without further knowledge. But having further knowledge would defeat the purpose of asking the question in the first place. With RAG, the LLM or the whole tool running the LLM can provide the user with the documents. The user can use these documents to verify the answer or gain additional insights.

Using all available documents would exceed the maximum sequence length of any LLM, even for a small domain. It is therefore necessary to implement a retrieval mechanism that selects the best documents for the query. The best documents are those that have a high chance of containing the answer or part of the answer. Different approaches exist, but the initial paper used Dense Passage Retrieval (DPR) for this. DPR converts all available sources into a vector representation using a transformer-based model. The same process is done with every query. Finally, the document vectors closest to the query vector in vector space are selected as context for the question. This is done by calculating the cosine similarity, or dot product, between the vectors. [Kar+20]

Closed-book QA is a difficult process due to the necessity of baking facts into parameters, like the previous approaches tried. RAG avoids this and enables open-book QA for every domain. Another key advantage of RAGs key advantage is its ease of use. The previous approaches required extensive preparation or training to gain domain-specific knowledge. The only necessary requirement for RAG is providing the data in some database and combining it with preexisting implementations of the retrieval process.

This thesis uses RAG with external models to compare against the models trained with this work.

The disadvantage of RAG is that the retrieval mechanism has to be implemented and managed. While this disadvantage is minor, it would still be preferable if any necessary information could be stored directly in the parameters of the model. This is especially true for domains with few changes over time. Another potential downside of RAG is the fact that the model never learned any domain-specific vocabulary during training. If the contexts contain a vocabulary the model has never seen, it is possible that the model doesn't see the connection and therefore fails to answer the question. This is also true if the provided context doesn't contain the answer at all, due to a low-quality retrieval process.



# 3

## Methodology

Two models were trained for this thesis using the same training scripts. One model was trained completely from scratch, while the other used an existing externally available model and only added domain-specific pretraining. Additionally, different fine-tuned variants of both models were trained to evaluate the effect of fine-tuning the models. This chapter describes the datasets used for pretraining and fine-tuning and the models trained with these datasets. It also describes the exact training setup and training process. This includes an overview of the used scripts and selected parameters.

### 3.1. Training Data

The data used for training is split into different parts. Pretraining and fine-tuning work with different datasets. As previously mentioned, administration data will be used during pretraining with the goal of increasing QA capabilities for administration questions. All data except the administration data is obtained directly from Hugging Face via the Transformer library.

#### 3.1.1. Pretraining datasets

As inspiration for training data, the German GELECTRA model was used [CSM20]. The bulk of the data consists of the OSCAR dataset. OSCAR is an open-source corpus that takes the multilingual Common Crawl corpus, classifies, and filters it. Common Crawl contains the raw web page data, metadata extracts, and text extracts of millions

of websites [Cra24]. While the whole Common Crawl contains petabytes of data, the current OSCAR monolingual German corpus contains about 145GB. This is due to filtering certain lines, deduplication, only using the raw text data of the website, and only using the German websites of the latest crawl. [SSR19] The dataset from Huggingface is from 2019.

The quality of text on an average website is very low. Websites that only contain sexual advertisements do not add significant information necessary for the underlying problem. To include some higher-quality and more consistent data, the German Wikipedia dump is also used. This adds 9GB of additional data from up to November 1, 2023, to the available datasets. The data is provided by Wikimedia, the company behind Wikipedia. Each entry consists of one filtered Wikipedia article. The filtering process excluded unwanted parts like markdown and undesirable sections like references. [Fac24a]. The dataset contains additional information about the URL and the title, but those columns were ignored for this thesis. Only the text columns of both the OSCAR and the Wikipedia datasets were used. GELECTRA used additional datasets for training, but neither of these were available via Hugging Face. Using these would have involved external downloading and preparation of these datasets. This doesn't scale well to other datasets and is therefore not preferable. [CSM20].

### 3.1.2. Administration Data

With about 6 MB of data, the administration data set is much smaller compared to the pretraining data set. However, with 2100 texts describing different areas of German administration, it describes all public services available in the German federal state of Baden-Württemberg. The information is split into two strongly connected categories: "Service" and "Living situation". Each document under the "Service" category describes a single service available at a public authority or related areas. An example would be a description of the process to register a new car, containing the competent authority, required prerequisites, and associated costs. All information groups (costs, prerequisites, etc.) are separated into different sections. The about 800 "Service" documents will be the main focus of QA evaluation. "Living Situation" is a broader term that doesn't have a single service but multiple possible applicable services attached to it. Additionally, it contains a description of necessary or useful information related to the situation. Staying with the previous example, "car" would be the corresponding living situation. In total, about 1300 different living situations are described in this dataset. The common way of accessing these documents is via the URL <https://www.service-bw.de/zufi>. [Bad24]

The whole dataset splits each document by section, leading to each entry contain-

ing one section with one piece of information from a single document. Each entry then starts with the header of the whole document to retain the necessary context information. This approach keeps the entries small, which is useful for the small contexts used in this thesis. The original dataset was in JSON format, which is in contrast to the required JSON Lines format for training. The JSON Lines format is a document that has a single JSON object in every line. Additionally, it was used for RAG and therefore included multiple extra fields like an embedding, which didn't contain any relevant training data for this thesis. The only required information was in the "text" field of each entry. Because of this, the JSON file was formatted beforehand to be a JSON Lines file containing multiple entries with only the single "text" field.

During this process, an error occurred that removed all ',' characters from the file. This is especially noticeable with prices, since in German prices have a ',' between the euro and the cent part. "93,00€" therefore becomes "9300€". This issue wasn't noticed until after pretraining, which made it impossible to fix. Nonetheless, as all evaluation models will use the exact same data, it should not affect the quality of the outcome; instead, it should alter the correct answer.

### 3.1.3. Fine-tuning datasets

A separate dataset is used for each fine-tuning task. The fine-tuning datasets are used for three different tasks: closed-book QA, open-book QA, and classification. Since no German dataset for closed-book QA exists, this work uses Deepset's GermanQuAD open-book QA datasets without context. To test different approaches, Deepsets GermanDPR was used for open-book or closed-book question answering, depending on the exact model. GermanQuAD contains about 13,722 different question-answer pairs, while GermanDPR adds 10,300 additional QA-pairs [MRP21]. For classification, germeval18 is used, which contains about 5,000 entries. [Ris+18].

- deepset/GermanQuAD: The GermanQuAD from deepset is inspired by Standfords SQuAD [Raj+16]. It also contains questions that are close to the English "Natural Questions" dataset, containing questions that are easily convertible to closed book questions [Kwi+19].
- deepset/GermanDPR: Is created in the same way as GermanQuAD, but with different questions. The main difference is that additional contexts are provided that don't contain the answers. This thesis ignores these contexts.
- philschmid/germeval18: Dataset, which labels different social media posts by their offensiveness. Their multi-label options include PROFANITY, INSULT, ABUSE, and OTHER.

### 3.2. Models

The models trained for this thesis follow two different approaches. The first approach just uses an existing model configuration to train the model from scratch. But training a model from scratch is a lengthy and expensive process, even for small models. As a result, a different, even cheaper option was also explored. This option used an already existing small German model as a checkpoint for training. Both models are T5 variants. The models can be found on the Hugging Face platform: <https://huggingface.co/dwolphers>.

#### 3.2.1. $T5_{BASE}$ - Pretrained model from scratch

The model from scratch has the same configuration as Google's t5-v1.1-base, containing about 250 million parameters. Differences from the original T5 description consist of a different activation function for the FNN and slightly more parameters by not sharing parameters between the embedding and classifier layers [Raf+21]. Additionally, the configuration used is a combination of the original base and small configurations, as described in the paper [Raf+19]. This configuration has a total of 24 transformer blocks, with 12 encoder and 12 decoder blocks. The following table shows an overview of all relevant configuration parameters: 3.1. This model and its variants will be called  $T5_{BASE}$  in this paper.

Table 3.1: Configuration parameters for  $T5_{BASE}$

Parameter	Value	Explanation
Encoder blocks	12	Number of encoder blocks
Decoder blocks	12	Number of decoder blocks
$d_{ff}$	2048	Output dimensionality of the hidden layers in the FNN
$d_{kv}$	64	Dimensionality of the query, key and value matrices
Number heads	12	Number of attention heads
$d_{model}$	768	Dimensionality of all other sub-layers and embeddings

This model uses all available datasets for pretraining. But due to a combination of limited training time and trying to increase the focus on the administration data, only

50% of the OSCAR and Wikipedia datasets are used. Additionally, to increase the importance of the administration dataset compared to OSCAR and the Wikipedia dump, a hundred duplications of the administration dataset are used during pretraining. This could result in a desirable overfitting, as mentioned in 2.2.2. In total, the following datasets were used to pretrain  $T5_{BASE}$  3.2.

Table 3.2: Datasets used to pretrain  $T5_{BASE}$ 

Dataset	Config	Factor
OSCAR	unshuffled_deduplicated_de	0.5
Wikipedia	20220301.de	0.5
Administration		100

### 3.2.2. $T5_{LARGE}$ - Use an existing German $T5$ model and add domain-specific pretraining

Table 3.3: Configuration parameters for  $T5_{LARGE}$ 

Parameter	Value	Explanation
Encoder blocks	36	Number of encoder blocks
Decoder blocks	36	Number of decoder blocks
$d_{ff}$	3072	Output dimensionality of the hidden layers in the FNN
$d_{kv}$	64	Dimensionality of the query, key and value matrices
Number heads	12	Number of attention heads
$d_{model}$	768	Dimensionality of all other sub-layers and embeddings

The second model relies on the pretrained GermanT5 "t5-efficient-gc4-german-base-nl36" with about 600 million parameters. Due to its larger parameter size, this model will be called  $T5_{LARGE}$  for this paper. This larger parameter count is a result of the higher number of decoder and encoder blocks used. In total, 72 blocks are used, with

36 decoder and encoder blocks, respectively. As seen in table 3.3, this is again a deviation from the original configurations, with a very high number of blocks, while the other parameters are equivalent to the original "base" configuration.

GermanT5 models were solely chosen because no other monolingual German T5 models were available. The base model was chosen because the total training time roughly aligned with the expected training time of  $T5_{BASE}$ . This should reduce any potential advantage for either side when it comes to total training time.

According to Hugging Face statistics, the model was trained for 524.288 steps over the course of five days [Ger22]. The exact hardware used isn't 100% clear. The hugging face model page mentions four V100 GPUs, but only for evaluation. On the other hand, their GitHub page describes a process for pretraining a T5 model, which uses GCP and a TPUV3-8. Since the second approach is precisely for pretraining, this paper assumes the later was chosen. [Sch22b]

The data used for pretraining was the GC4, a German colossal, cleaned Common Crawl corpus. Just like OSCAR, the GC4 is a German split of the Common Crawl dataset [Cra24]. GC4 splits the data into three categories [May24]:

- HEAD: Consists of high-quality text (e.g., newspapers, government websites)
- MIDDLE: More colloquial language like forum entries and commentary sections
- TAIL: The dark side of the Internet

GermanT5 only used the high-quality data from the HEAD to perform training, which was already a raw 286GB of data. Just like in this thesis, the tokenizer used was a SentencePiece unigram tokenizer. To use this model, the model had to be converted to its Flax representation, which could easily be done with Hugging Face. [Sch22b]

Using this model, a new model was trained by adding additional pretraining steps with the administration data. Just like for  $T5_{BASE}$ , the effect of the administration data was increased by duplicating the administration dataset a hundred times, resulting in the following datasets: 3.4.

Table 3.4: Datasets used to pretrain  $T5_{LARGE}$

Dataset	Config	Factor
GC4	HEAD	1
Administration		100

While the larger model might still have an advantage due to its larger size, this shouldn't affect the results too much. As highlighted in 2.2.1, more parameters only really result in better answers when the training time is adjusted accordingly. The goal of this model is to see if it is possible to skip the generic pretraining step and rely on externally pre-trained models instead. If possible, this approach could easily scale to larger models, in contrast to the other approach.

### 3.2.3. Variants

To evaluate the effect of fine-tuning on the models, different approaches were tested. These approaches were identically applied to  $T5_{BASE}$  and  $T5_{LARGE}$ .

Since the key incentive for this thesis is closed-book QA, the first fine-tuning step was always closed-book QA with GermanQuAD. The first approach reinforces this aspect by adding an additional closed-book QA fine-tuning step with GermanDPR. The models trained with this approach will be called  $T5_{BASE\_CLOSED}$  and  $T5_{LARGE\_CLOSED}$ , respectively.

This paper also wants to explore the effect of domain-specific pretraining on open-book QA. Can small LLMs with domain-specific pretraining gain the same results as open-book QA in the respective domain compared to larger LLMs? To incorporate this, the second approach uses GermanDPR for an open-book QA fine-tuning step. Analogous to the first approach, the resulting models will be called  $T5_{BASE\_OPEN}$  and  $T5_{LARGE\_OPEN}$ .

The final fine-tuning approach wants to test the effect of a seemingly unrelated generic NLP task on the results of the answers. The goal is to see if the quality of answers for QA rises when a completely different task and dataset are added at the end. To achieve this, all previously fine-tuned models are fine-tuned with the germeval18 dataset on classification. The resulting models will be named with a *CLASS* suffix in this paper. This results in the following variants 3.5:

## 3.3. Hardware and GCP Setup

For training, Google Cloud had to be set up with the required hardware and storage. This section describes the decision-making process for these requirements. The script "setup\_google\_cloud.sh" contains the necessary commands to setup GCP with the exact same setup via the API.

Table 3.5: Datasets used to fine-tune all T5 variants

Variant	GermanQuAD Closed Book	GermanDPR Closed Book	GermanDPR Open Book	Germeval18
$T5_{BASE\_CLOSED}$	x	x		
$T5_{BASE\_CLOSED\_CLASS}$	x	x		x
$T5_{BASE\_OPEN}$	x		x	
$T5_{BASE\_OPEN\_CLASS}$	x		x	x
$T5_{LARGE\_CLOSED}$	x	x		
$T5_{LARGE\_CLOSED\_CLASS}$	x	x		x
$T5_{LARGE\_OPEN}$	x		x	
$T5_{LARGE\_OPEN\_CLASS}$	x		x	x

### 3.3.1. TPU and CPU selection

The TRC programme for this thesis allowed access to three different TPU types. In the zone "europe-west4-a" up to five simultaneous on-demand TPUv3-8 devices could be used. Zone "us-central1-f" offered an additional five simultaneous on-demand TPUv2-8 devices. Both of these were used for this thesis. GCP also offers preemptible TPUs for a cheaper price, with the downside that these can be shut down at any point. With the TRC programme, a hundred preemptible TPUv2-8 devices in "us-central1-f" could be accessed for free. Due to the large downside, these were not used for this work.

The v3-TPUs were used for pretraining of  $T5_{BASE}$ . This exploits the performance improvements of the version 3 TPUs as described in the section 2.3. V2-TPUs were used to train  $T5_{LARGE}$  due to the low pretraining requirement. No working training script to fine-tune T5 on a TPU was found for this thesis. While it would probably be possible to create one, that was not the goal of this thesis. Since fine-tuning requires far less computational power, this wasn't a large issue. Instead, an available script using CPUs was used with a local and available Intel® Core™ i9-12950HX processor. A "c2-standard-30" GCP-VM would offer a similar CPU for about 1,59\$/h. This estimate will be used to calculate the costs of fine-tuning in the next chapter.



### 3.3.2. Virtual Machine

The TRC programme was only available with TPU-VMs, restricting the possibility for this thesis to explore possible advantages for different TPU access methods. This shouldn't be too much of an issue, as Vortex AI doesn't support Flax and GKEs flexible scaling wasn't necessary for this thesis.

To create an VM, a base image has to be selected, containing the operating system and preinstalled packages. All images have Python preinstalled, which is the main programming language for ML. Additionally most of these images come with preinstalled common ML libraries like TensorFlow or PyTorch. This is not the case for Flax or JAX, where no images exist that have the libraries preinstalled. Instead these frameworks have to be installed manually with python. Due to this the requirements for the selected base image were very low. Instead an option could be chosen that avoids possible version conflicts between libraries by having close to none installed. Because of this, "v2-alpha" was chosen, but other options were available as well. It is unlikely that the choice of the image affected the training process in any way.

### 3.3.3. Storage

Storage requirements are the last factor to consider during the setup for GCP. The two elements requiring most of the data are the model with all saved checkpoints and the training data. Due to the small models used for this thesis and no additional checkpoints saved during pretraining, model checkpoints make up only a small percentage of the space requirement. Due to this, the model can be saved directly on the VM. This has the downside of deleting any saved model data when the VM gets destroyed. The low size of the model again lessens this issue, since any model data can easily be pulled from the VM beforehand.

For training data, this is not the case. OSCAR's 140GB dataset is much larger than the model, and downloading or uploading this data for every process is undesirable. This is especially true if multiple approaches with the same training data are run. Additionally, training data has to be prepared for the training steps. During this process, intermediate steps of the mapping process get saved to the disk. This caching drastically improves performance for any further re-usage of the same dataset. But it also increases the required data storage even further. External persistent storage is a relatively cheap option to circumvent this issue (a TB of data costs 400\$ for a whole month). Training  $T5_{BASE}$  in "europe-west4-a" required all available datasets. In addition, different training approaches with the test data were used here, which led to more caching space being necessary. Consequently, to avoid any possible issue with storage, the data disc

for the VM in "europe-west4-a" was 1200GB.

For "us-central1-f" pretraining was only done with the administration data, and fine-tuning wasn't done at all. Due to testing with different approach some external storage was still useful, but in this case the disc size was only 500GB. This has the additional benefit of incorporating future fine-tuning storage costs into the cost evaluation.

### 3.4. Training Process

The training process is the main part of this thesis. It relies heavily on t5-flax-gcp, which itself uses multiple scripts and tools provided by Hugging Face [SN22]. For fine-tuning, multiple slightly altered variations of the FLAN-t5's fine-tuning script were used, which also rely on the Hugging Faces Transformer framework [Tou23]. All training scripts are Python files or shell scripts running these Python files. All scripts can be found on GitHub (<https://github.com/da721wol/Domain-specific-LLM>). The repository is a fork of the original t5-flax-gcp: <https://github.com/gsarti/t5-flax-gcp>.

It is important to note that the only connection to the VM is via SSH. It is possible and likely that the SSH connection might stop for a short time during long processes. Running any long process should therefore always be done in the background and not in the terminal. Otherwise, the process will crash as soon as the SSH connection disconnects. This is particularly true for the long-running Python training processes. To run them in the background, both "nohup" and "&" are required 3.1:

Listing 3.1: Example background call for a Python file

```
nohup python3 file.py > output.log &
```

#### 3.4.1. Tokenizer

T5<sub>BASE</sub> and all variants use the same self-trained SentencePiece Unigram tokenizer. T5<sub>LARGE</sub> and its variants rely on the tokenizer trained during pretraining for GermanT5. The vocabulary size of the self-trained tokenizer was 31.000. The training was conducted on the initial 100.000 entries from the OSCAR dataset [SSR19]. Since SentencePiece removes the necessity of pretokenizing the input, raw input sentences can be passed to the training script. Given the deterministic nature of the training, while also having a minimal potential impact of using varying vocabularies, the tokenizer was trained only once.

The training code for the tokenizer is a copy of DeDLOC’s tokenizer implementation [DeD21], which was also used by HuggingFace [Fac24b]. It represents a custom SentencePiece unigram tokenizer with lower-casing, digits, punctuation, and Bengali character normalization. The script “create\_tokenizer\_cfg.py” contains this code and has a few required parameters, which are listed in the table 3.6.

Table 3.6: Arguments for the tokenizer script

Parameter name	T5 <sub>BASE</sub>
model_dir	<Path to where the tokenizer should be saved to>
dataset	oscar
dataset_config	unshuffled_deduplicated_de
dataset_split	train
text_field	text
vocab_size	31000
input_sentence_size	100000
config_type	google/t5-v1_1-base

The exact Python call can be found in “train\_t5\_model.sh”. This includes the installation of all required Python packages with the provided “requirements\_pretraining.txt” file.

### 3.4.2. Pretraining

Pretraining was conducted with a slightly modified version of a Hugging Face script, called “run\_t5\_mlm\_flax.py”. As the name suggests, Flax was used for loading and training the model, while “mlm” refers to MLM, the same process described in Section 2.2.3. Two adaptations had to be done, one for T5<sub>BASE</sub> and one for T5<sub>LARGE</sub>. The original script only accepted a single Hugging Face dataset or a local dataset from a file. For this thesis, the script was adapted to accept any combination of up to two Hugging Face datasets and a local dataset. This allows all training datasets for T5<sub>BASE</sub> to be passed separately.

For T5<sub>LARGE</sub>, the initial training showed infinite loss and NaN (not a number) for accuracy. Further testing showed that this was a result of the previous training being done in

a different floating-point format. As mentioned in Section 2.3.1, TPUs support FP32 and BFLOAT16. Converting the parameters to BF16 fixed the issue and enabled training.

The training script accepts a long list of arguments. To order them, the script splits arguments into three different classes for different use-cases.

- **ModelArguments:** Describe the model and tokenizer trained with this script. This can either be an existing pretrained model or just the configuration to train a new model from scratch.
- **DataTrainingArguments:** Arguments that describe the data used for training. It also contains parameters that are unique to MLM, like the masking probability.
- **TrainingArguments:** Contain the generic arguments required for all training setups. This class mostly consists of hyperparameters.

The following table 3.7 shows an overview of the required parameters and their values. Values that use a local path contain a description instead. While no hyperparameters are required (they all have default values), the last section 3.4.4 of this chapter will give an overview of the selected hyperparameters for training.

Table 3.7: Arguments for the pretraining script

Argument class	Parameter name	T5 <sub>BASE</sub>	T5 <sub>LARGE</sub>
ModelArguments	model_type	t5	t5
	config_name	google/t5-v1.1-base	
	model_name_or_path		<Path to the saved GermanT5 model>
	tokenizer_name	<Path to the saved tokenizer>	<Path to the saved GermanT5 model, which should contain the tokenizer>
DataTrainingArguments	dataset_name	oscar	
	dataset_name2	wikipedia	
	dataset_config_name	unshuffled_deduplicated_de	
	dataset_config_name2	20220301.de	
	train_file	<Path to the local domain-specific data>	<Path to the local domain-specific data>
TrainingArguments	output_dir	<Any folder to save the trained model to>	<Any folder to save the trained model to>
	adafactor	True	True
	preprocessing_num_workers	96	96

Just as for the tokenizer, the exact call can be found in "train\_t5\_model.sh".

### 3.4.3. Fine-tuning

As mentioned in Section 3.3.1, fine-tuning was processed on a CPU. The fine-tuning scripts were slight variations of an available fine-tuning script for FLAN-T5. FLAN-T5 is an English T5 model that has been fine-tuned for more than 1000 additional tasks. [Chu+22] The available script ([Tou23]) was focused on open-book QA but could easily be adapted for other fine-tuning tasks. The resulting scripts are called `finetuning_t5_<FINE-TUNING_TASK>.py`. Fine-tuning requires some additional and altered Python libraries, which can be installed with the provided `"requirements_finetuning.txt"` file.

To keep the fine-tuning process as close to T5 as possible (as described in 2.2.4), all datasets were slightly altered. For each task, a different prefix is added to the input:

- Closed book QA: "Beantworte die Frage: "
- Open book QA: "Beantworte mit folgendem Kontext die Frage: "
- Classification: "Bewerte folgende Aussage: "

For open-book QA, the question and the context are also merged together to produce a single input for the fine-tuning process. The resulting input was: "Beantworte mit folgendem Kontext die Frage: {question} {context}". Additionally, the labels were defined by selecting the expected answers from the dataset. Both input and labels were then run through the tokenizer.

The fine-tuning process relies on the `Seq2SeqTrainer` class from the `Transformers` library. This trainer again uses multiple required or optional parameters, most of which are hyperparameters. Again, a detailed overview of hyperparameters will be given in the next section, 3.4.4. Most `TrainingArguments` for the `Trainer` class focus on activation or deactivation of different aspects, which is why the following table 3.8 will focus on explaining parameters for the `Trainer` class.

### 3.4.4. Hyperparameters

The hyperparameters used for this thesis rely on a combination of the predefined parameters from `t5-flax-gcp` and the parameters described in the paper for GELECTRA [CSM20]. Due to the expected low impact and high relative cost of hyperparameter optimisation (as mentioned in 2.2.2), no extra optimisation was done. The only exception to this is the decreasing of the learning rate over time, which is integrated into the `AdaFactor` optimizer. For `T5LARGE`, the hyperparameters were read from their available configuration files that were used for their pretraining example [Sch22a]. It is very likely

Table 3.8: Arguments for fine-tuning scripts

Parameter	Explanation
<code>output_dir</code>	The folder to save the trained model to
<code>predict_with_generate</code>	Required for the evaluation process with ROUGE used in this script. More detail on ROUGE in Section 4.2
<code>push_to_hub</code>	Whether to push to Hugging Face, requires being logged in with the <code>huggingface-cli</code>
<code>report_to</code>	Training and evaluation results can be reported to track performance. Possible options include "wandb" or "tensorboard".
<code>use_cpu</code>	Force use of CPU
<code>generation_max_length</code>	Avoid using the default max length for generation during evaluation, since this is only 20 tokens

that these parameters were also selected for the training of "t5-efficient-gc4-german-base-nl36". An overview of the selected pretraining hyperparameters can be seen in the table 3.9. For the additional training steps of  $T5_{LARGE}$ , smaller hyperparameters had to be chosen due to the low size of the dataset (even with duplicated data). Due to limited memory on the TPUs, the chosen batch size had to be lower than the official recommendations from Google (a multiple of 64) for both models.

For fine-tuning, the hyperparameters for the different tasks were very similar. The only exceptions were sequence length and batch size. To increase training speed, the sequence length was reduced for all tasks that never reached the maximum sequence length anyway. For tasks that required the maximum sequence length, batch size had to be reduced to avoid performance issues due to memory limitations.  $T5_{LARGE}$  was already selected due to the similar training time compared to  $T5_{BASE}$ . To keep this consistent, the number of training epochs was cut in half for  $T5_{LARGE}$ . This should cancel out the effect of longer calculations for the larger model to some extent and result in similar training times. Additionally it can show the result of less fine-tuning for the domain-specific evaluation. An overview of the selected fine-tuning hyperparameters can be seen in the table 3.10.

Table 3.9: Hyperparameters for pretraining

Hyperparameter	$T5_{BASE}$	$T5_{LARGE}$	$T5_{LARGE}$ Administration Training
Max sequence length	512	1024	512
Training batch size	32	16	8
Evaluation batch size	32	16	8
Training epochs	3	Not available	10
Warmup steps (k)	10	10	0.1
Learning rate	0.004	0.004	0.004
Checkpoint every (k)	100	50	1
Training steps	823,100	524,288	26,700
Eval every	2500	100	300
weight decay	0.001	0.001	0.001

Table 3.10: Hyperparameters for fine-tuning

Hyperparameter	Closed-book QA	Open-book QA	Classification
Max sequence length	128	512	128
Batch size	16	8	16
Training epochs ( $T5_{BASE}/T5_{LARGE}$ )	6/3	6/3	6/3
Learning rate	3e-04	3e-04	3e-04
Weight decay	0.01	0.01	0.01
Checkpoint every	epoch	epoch	epoch

# 4

## Results and Evaluation

This chapter will give an overview of the accumulated training time and the estimated costs. It will also highlight how the models trained in this thesis perform for different common NLP benchmarks to give a perspective on their general understanding of the German language. Finally, it will describe and conduct the evaluation process for domain-specific QA. Additional external models will be used as references to achieve this.

### 4.1. Training Results

The training results only present the training costs without any additional testing. While this underestimates costs for most use cases, it should be enough to give a general understanding of the expected costs associated with the different training processes.

#### 4.1.1. Training duration

The pretraining of  $T5_{BASE}$  was conducted over 823,100 steps in about 8 days.  $T5_{LARGE}$  had already been pretrained for 524,288 steps over the course of about 5 days. The 26,700 extra training steps took about 10 hours on the slower TPUv2-8.

The table 4.1 shows that fine-tuning the larger model had noticeable slower times over all tasks. But even while running fine-tuning on a much slower CPU compared to the TPUs, fine-tuning is still only a small margin of the total training duration for



$T5_{BASE}$ . For  $T5_{BASE}$ , fine-tuning is only about 10% of the total training time. In contrast,  $T5_{LARGE}$  this was not the case, as fine-tuning was responsible for about 80% of the total training duration. As this was only done with three epochs compared to  $T5_{BASE}$ 's six shows, that externally trained models have a much higher focus on fine-tuning.

The fine-tuning statistics also contained `total_flos`, the total number of floating-point operations that were executed during fine-tuning. This enables a comparison with TPUs, which is why these numbers were included as well. This is also a fixed value, which doesn't depend on any other work done on the local machine during training.

The duration of the fine-tuning depends on two key factors: the number of training steps and the complexity per training step. Just like in pretraining, the number of training steps is a combination of the dataset size and the number of epochs. The complexity of a training step is mainly defined by the number of tokens in each step. This can be seen in the training results. Open-book QA has a high number of tokens due to the provided context and is consequently by far the most costly fine-tuning. And closed-book GermanQuAD is more expensive than GermanDPR since the dataset is slightly larger.

Table 4.1: Training time

	Training Time (in hours / $10^{12}$ FLOS)	$T5_{BASE}$	$T5_{LARGE}$
TPU	Total previous pretraining	0	124
	Total additional pretraining	192	10
CPU	Closed-book QA - GermanQuAD	4 / 2,528	6 / 2,576
	Closed-book QA - GermanDPR	2 / 1,887	4 / 1,924
	Open-book QA - GermanDPR	13 / 26,570	26 / 36,258
	Classification - germeval18	3 / 3,083	4 / 3,043

#### 4.1.2. Cost

The main cost factor in training is the TPU runtime. Since this work was accepted for the TRC, one month of TPU time could be used free of charge. Nevertheless, the runtime can give an estimation of the expected costs. The price for TPUs depends on the

version and the region (each region consists of multiple zones, e.g., "europe-west4" with "europe-west4-a", "europe-west-4b", etc.), but the version is the main price-determining factor. TPUv2 in "us-central1" is therefore cheaper than TPUv3 in "europe-west4", with 1.50\$ per chip-hour compared to 2.00\$ for a v3. Both TPUv2-8 and TPUv3-8 have four chips, which results in a total cost per hour of 6.00\$ and 8.00\$, respectively.

For storage, the price per GB is very close, with 0.044\$ per month for "europe-west4-a" and 0.04\$ per month for "us-central1-f". The main difference in price is, therefore, the size of the storage. 1200GB of storage in "europe-west4-a" cost 52.80\$ per month or about 0.073\$ per hour, while 500GB of storage in "us-central1-f" cost 20.00\$ per month or about 0.028\$ per hour.

Table 4.2: Training costs

Training Costs (in \$)		T5 <sub>BASE</sub>	T5 <sub>LARGE</sub>	1200GB Storage	500GB Storage
TPU-costs/CPU-Costs or Storage-Costs		8\$/h/1.59\$/h	6\$/h/1.59\$/h	0.073\$/h	0.028\$/h
TPU	Total additional pretraining	1536.00	60.00	14.80	0.28
CPU	Closed-book QA - GermanQuAD	6.36	9.54	0.07	0.17
	Closed-book QA - GermanDPR	3.18	6.36	0.07	0.11
	Open-book QA - GermanDPR	20.67	41.34	0.59	0.73
	Classification - germeval18	4.77	6.36	0.15	0.11
Total costs		1570.98	123.60	15.62	1,40

Since pretraining costs for T5<sub>LARGE</sub> were much cheaper, the total costs are also drastically different, as seen in the table 4.2. Even while using less cost-efficient CPU-prices, pretraining still led to over 97% of the total training costs for T5<sub>BASE</sub>. In total, training a model from scratch was about twelve times more expensive than using a pretrained model. And this doesn't even consider the higher parameter count and the consequently longer fine-tuning duration. Ignoring any fine-tuning, the difference goes up to factor 25.

Storage costs only make up a small amount of the total costs and can therefore be ignored for most use-cases. The same can be said about networking traffic costs, which had a total cost of only 1.50\$ for both models for this thesis. This was mainly due to the low number of checkpoints saved externally. During pretraining, all model checkpoints were saved on the disc, and none were pushed to external storage options like Hugging Face. Only the final results of pretraining and fine-tuning were pulled from the VM, resulting in low costs. Saving all checkpoints externally, having a high number of checkpoints, and having a larger model would all result in higher network costs. But

even with these factors included, the main cost item would remain pretraining when training a model from scratch.

Using an external model results in a more balanced cost distribution. While storage and network costs remain relatively low, the focus on fine-tuning costs increases. For  $T5_{LARGE}$ , this is basically a 50/50 split.

#### 4.1.3. Availability

During the whole free month, no larger TPUs-Pods (with 8 or more chips) were available. It was therefore not possible to test any other configuration or larger models. TPUV2 were easily available, but this is very likely because the zone for these TPU versions is only available for TRC users. Any non-research approach will therefore not be able to use this zone. Additionally, TPUV2 pods were not available in "us-central1-f", blocking this approach as well.

Access to TPUV3 in "europe-west4-a" was also limited. To gain access for this thesis, a script was used that checked for an available TPUV3-8 every ten minutes. After 3 days, this script resulted in a successful setup VM with a TPUV3-8. The same script was used for a month to gain access to a TPU-Pod without success. This script was integrated into "setup\_google\_cloud.sh".

## 4.2. Evaluation results for common NLP benchmarks

While answering domain-specific questions is the focus, general language understanding is also necessary to achieve this goal. Should the models consistently underperform on these benchmarks, it is highly probable that this will translate to subpar results for domain-specific questions as well. Consequently, any exceptional low-performing variants were excluded from further evaluation. To evaluate the general ability to understand the German language, common NLP benchmarks are conducted on the different models. For evaluation, the test splits of the fine-tuning datasets (3.1.3) are used.

Evaluation is done with the ROUGE (Recall-Oriented Understudy for Gisting Evaluation) score, a score that compares n-grams between the predicted answer of the model and the provided answer (or a possible list of answers). The score ranges between 0.0 and 1.0, where higher scores indicate better matching [Chi04]. While ROUGE was initially used for evaluation of summarising tasks, it was shown that ROUGE can also be utilised to evaluate QA tasks [Che+19]. This thesis uses ROUGE1 and ROUGE2 for evaluation, which compare unigrams and bigrams, respectively. Additionally, ROUGEL was used,

which uses the longest common subsequence.

#### 4.2.1. Open-Book Question and Answering

The evaluation for open-book question answering was done for both the GermanQuad and GermanDPR to show the quality of answers for different datasets. Table 4.3 shows the scores for the non-classification variants. The external pretrained model beats the own pretrained model for GermanQuaD, but losses against  $T5_{BASE}$  for GermanDPR. In both cases the differences are very low, showing that both models have similar general open-book capabilities.

Additionally, scores for the GermanDPR dataset are about 0.1 higher than the German-Quad dataset for both models. As previously mentioned, open-book fine-tuning was only done on the GermanDPR dataset, but general open-book question-answering capabilities should be derived from this. GermanDPR's higher focus on document retrieval questions might also impact the scores. It's therefore possible that the questions are just more fit to be answered with open-book QA.

Looking at the variants fine-tuned with an additional classification step shows that these models are completely overfitted for the classification task. Checking the actual answers shows that basically all questions get answered with 'OTHER', one of the labels for classification. Even  $T5_{LARGE}$  with only three epochs doesn't show any noticeable improvements.

Table 4.3: Open-book QA - NLP Benchmarks

Dataset	ROUGE-Score	$T5_{BASE.OPEN}$	$T5_{LARGE.OPEN}$	$T5_{BASE.OPEN.CLASS}$	$T5_{LARGE.OPEN.CLASS}$
GermanQuad	ROUGE1	0.512	0.519	0.002	0.003
	ROUGE2	0.378	0.386	0.001	0.000
	ROUGEL	0.510	0.518	0.001	0.003
GermanDPR	ROUGE1	0.624	0.600	0.005	0.004
	ROUGE2	0.480	0.457	0.002	0.000
	ROUGEL	0.622	0.596	0.005	0.004

#### 4.2.2. Closed-Book Question and Answering

While closed-book QA on domain-specific questions is the focus of this thesis, fine-tuning was also done for general questions. Again, both the GermanQuad and the

GermanDPR datasets are used for this. As shown in Table 4.4, both models perform badly on closed-book questions in general. Especially for GermanQuad, the scores are extremely low, but again, with a very slight but constant advantage for  $T5_{LARGE}$  compared to  $T5_{BASE}$ . The scores for GermanDPR are slightly better, with a larger advantage for  $T5_{BASE}$  this time. Since the scores for open-book QA were already better for GermanDPR, it seems that the questions from GermanDPR are either "easier" for an LLM to answer or the test dataset is closer to the training data set.

As already noticed in 4.2.1 the classification variants again perform terrible. This is notable, since fine-tuning for closed-book QA was done with both datasets, resulting in twice as much fine-tuning. Nevertheless classification fine-tuning seem to have been enough to overfit both models.

In general, these low results are expected since both models and their variants were never expected to gain any general knowledge. The key goal of this fine-tuning step was never to actually answer a generic question but to understand the concept of a question. The next section will show if this results in better results for domain-specific questions.

Table 4.4: Closed-book QA - NLP Benchmarks

Dataset	ROUGE-Score	$T5_{BASE.CLOSED}$	$T5_{LARGE.CLOSED}$	$T5_{BASE.CLOSED.CLASS}$	$T5_{LARGE.CLOSED.CLASS}$
GermanQuad	ROUGE1	0.082	0.088	0.012	0.002
	ROUGE2	0.014	0.014	0.002	0.000
	ROUGEL	0.078	0.084	0.012	0.002
GermanDPR	ROUGE1	0.235	0.166	0.025	0.002
	ROUGE2	0.095	0.047	0.006	0.000
	ROUGEL	0.229	0.163	0.025	0.002

### 4.2.3. Classification

4.2.1 and 4.2.2 suggest that all classification variants seem to overfit on the classification task. To evaluate if this actually resulted in strong classification NLP-benchmarks, an additional evaluation was conducted. The classification evaluation was done on the germeval18 dataset and only with the ROUGE1 score. Since the answer is always a single-word label, the other scores wouldn't add any additional knowledge. The table 4.5 therefore only shows the ROUGE1 scores for all classification models for the germeval18 dataset.

Table 4.5: Classification - NLP Benchmarks

Dataset	ROUGE-Score	T5 <sub>BASE_CLOSED.CLASS</sub>	T5 <sub>LARGE_CLOSED.CLASS</sub>	T5 <sub>BASE_OPEN.CLASS</sub>	T5 <sub>LARGE_OPEN.CLASS</sub>
germeval18	ROUGE1	0.742	0.740	0.737	0.744

While these numbers are pretty high, they can be misleading. 66% of the classification dataset has the label 'OTHER', which is the go-to answer for the models after fine-tuning, as mentioned in the previous section. This makes the task much easier than the other tasks; therefore, a higher score is expected.

So while these variants seem to be exceedingly overfitted on classification tasks, as evidence in sections 4.2.2 and 4.2.1 suggests, this overfitting does not result in particular high scores. In combination with the previous scores, this indicates that classification variants actually lose their ability to understand questions while not gaining desirable benefits. Therefore, the evaluation of domain-specific QA would likely not result in any additional insight. This was confirmed during testing, showing that all answers are 'OTHER'. Consequently, all model variants with the *CLASS*-suffix will be excluded from further evaluation.

### 4.3. Evaluation of domain-specific questions

The previous section showed that both models and their non-classification variants have a general understanding of the German language. The only exception is closed-book QA, where all variants scored very low. This section will try to see if the answers to domain-specific questions have better quality. Both closed- and open-book QA will be compared to see if the models of this thesis produce better results for any type. Two additional external LLMs of different sizes will be used as a reference point.

#### 4.3.1. Process

The ROUGE metric faces challenges when evaluating answers to domain-specific questions without context. One key problem is that ROUGE needs predefined correct answers to score against, but often in specialised topics, there isn't just one right answer. This means the ROUGE score can vary greatly depending on which answers were chosen as the standard, even though different answers might mean the same thing. For example, even if two answers like "It's free" and "There's no charge" mean exactly the same, they don't share any common words, which can skew the ROUGE score and

misrepresent the correctness of the answers provided. While it is possible to increase the number of acceptable answers, it is likely that some variants will still be overlooked. To circumvent this issue, a different approach was used for the evaluation of the administration questions. First, a list of possible common questions was set up. Afterwards, a combination of the models trained for this thesis and external models with open-book QA were used to answer these questions. As mentioned in 2.4.3, RAG is currently the go-to approach to domain-specific QA. Open-book QA is the second step of the RAG process. The retrieval process is ignored to avoid including the quality of the retrieval mechanism for evaluation. Instead, the necessary context was hand-picked beforehand. The context contains the exact same document that was used for pretraining. The missing ‘,’ issue highlighted in 3.1.2 should therefore not result in different answers. It is important to note that only the `T5BASE_CLOSED` and `T5LARGE_CLOSED` models were subjected to closed-book style questions. All other models were never trained on closed-book QA or administration data. It would therefore be impossible for these models to answer the questions without context. All QA pairs can be seen in Appendix A.

The resulting QA pairs are then passed to human evaluators. They were asked to evaluate the answers according to a five-point scale:

- 0: The answer didn't fit the question at all. For example, a question about costs gets answered with a date.
- 1: The answer fits the question, but it's obviously wrong. For example, a question about the length of a process gets answered with multiple years.
- 2: The answer fits the question, but it's not obviously wrong. For example, a question about the length of a process gets answered with three weeks, while the true answer is three days.
- 3: The answer is either partly correct or misses some relevant information. For example, a question about the length of a process gets answered with the duration of a special express process but not the normal duration.
- 4: The answer is fully correct and includes most or all necessary information.

This scale is still subjective to some extent, which is why five different evaluators assessed the QA pairs. This should result in a general overview of the quality of the answers that one can expect from the different models.

### 4.3.2. Questions

The selected questions were chosen as likely questions a German citizen might want to know the answer to. Four different categories of question types were predefined to have a variety of possible questions. The chosen types were:

- *Time or money-related question*: If the context contains a straight-forward answer, the answer to these questions should include an exact figure for time, duration, or money. If this is not the case, the answer should explain why. An example would be: "It depends on something". The provided context can be longer when the exact figure depends on multiple different aspects.
- *Location or source*: The answer should contain a location or source. Sometimes a few aspects can lead to different locations, which should be mentioned. Nevertheless, the context is always short for these kinds of questions.
- *Yes/No-Questions*: These questions can be answered with a short yes or no, but longer answers are acceptable as well. The context length has a high variety for these questions.
- *Explanations*: The question requires some form of explanation of a process or word. Both the answer and the context for these questions can become very long.

### 4.3.3. Candidate models

The candidates chosen should represent a wide variety of possible options. GPT-4 was used for high-quality answers, while models from Aleph Alpha were used as a more common everyday LLM size. Both models were provided with the necessary context to answer the question.

#### German T5 models

The models used for evaluation were all trained during the training process. All models that were never fine-tuned on open-book QA were asked the question without any context. This included T5<sub>BASE\_CLOSED</sub> and T5<sub>LARGE\_CLOSED</sub>. In contrast, models fine-tuned with open-book QA were given the context. This included T5<sub>BASE\_OPEN</sub> and T5<sub>LARGE\_OPEN</sub>. The input queries were setup in the same form as for fine-tuning (3.1.3), as seen in 4.1 and 4.2.

Listing 4.1: Input Query T5 models closed-book

```
Beantworte die Frage: {question}
```



Listing 4.2: Input Query T5 models open-book

```
Beantworte mit folgendem Kontext die Frage: {question} {context}
```

The answers were generated with the Hugging Face Transformers library. The code can be found in the script "generate\_domain-specific\_answers.py".

### OpenAI - GPT-4

OpenAI is one of the leading LLM providers worldwide. They are most famous for ChatGPT, the chatbot that started the current hype around LLMs [Ope22]. A range of models developed by OpenAI can be accessed via the Chat-GPT interface. The latest of these is GPT-4, which is one of the best available generic LLMs at the moment [Ach+23]. For example, papers like [Wu+24] show that GPT-4 beats several other LLMs by a lot when it comes to multiple-choice questions. OpenAI showed in their own paper that GPT-4 can even exceed most humans for common large-scale tests like the Law School Admission Test (LSAT) or the Uniform Bar Exam [Ach+23]. While the exact number of parameters for GPT-4 is unknown, most estimates propose a figure around 1.7 trillion parameters [Lub23] [Bas23]. Consequently, GPT-4 serves as a benchmark for the highest quality of answers to compare to. GPT-4 should also show if certain questions pose a challenge even for high-quality models.

In contrast to the models trained for this thesis, GPT-4 doesn't need an exact template for QA. It was therefore possible to setup the input query in exactly the same format as in 3.1.3 (see 4.3). While further prompt engineering could probably increase the quality of answers further, the goal of this paper is to compare the raw performance for QA [HK23].

Listing 4.3: Instruction Template GPT-4

```
Beantworte mit folgendem Kontext die Frage: {question} {context}
```

### Aleph Alpha - luminous-base-control

GPT-4 operates on scales that even most larger companies will never reach. As a more achievable smaller LLM, Aleph Alpha was used as the second external model. Aleph Alpha is a German company focused on providing LLMs for companies and governments. Their target group overlaps with the example data from this thesis, which is why they were used as an evaluation candidate.

The exact model used was luminous-base-control. The luminous family are just the available LLM models from AlephAlpha, and with about 13 billion parameters, the base version is the smallest accessible. The control models are the fine-tuned variants, optimised to follow instructions. These models aren't as general as GPT-4, so just asking

the question with a context won't return any useful answer. Instead, the instruction template provided by Aleph Alpha for open-book QA was used. The models were likely fine-tuned on this structure, just as in 3.1.3. For the instruction part, the same sentence as in fine-tuning the GermanT5 models was used, resulting in the following template: 4.4.

Listing 4.4: Instruction Template AlephAlpha

```
### Instruction:
Beantworte mit folgendem Kontext die Frage.

### Input:
Kontext: {context}
Frage: {question}

### Response:
```

#### 4.3.4. Evaluation Results

Looking at the average results in the table 4.6, some observations can be made:

Table 4.6: Average Results for domain-specific Evaluation

Average-Scores	Closed-book		Open-book			
Question Type	T5 <sub>Base.Closed</sub>	T5 <sub>Large.Closed</sub>	T5 <sub>Base.Open</sub>	T5 <sub>Large.Open</sub>	luminous-base-control	GPT4
Time or money-related questions	2.1	1.2	3.0	2.5	2.0	3.8
Location or source	2.2	2.2	3.0	3.7	2.8	4.0
Yes/no-Questions	1.2	0.7	2.8	2.8	3.1	4.0
Explanations	1.0	0.7	2.1	2.5	2.8	4.0
Total	1.6	1.2	2.7	2.8	2.6	4.0

- GPT-4 is close to perfect: GPT-4 gets either a perfect 4.0 at every answer or is close to it. All answers, therefore, include all the relevant and correct information. Looking at the actual answer (as seen in Appendix A) shows that this is partly because GPT-4 gives extremely long answers. Answers like in 9 just include most of the context. In some cases, this is actually far more information than the average person asking the question would like to know. Since information overload was not part of the evaluation scale as described in 4.3.1, this doesn't show up in the results.

- Open-book beats closed-book: All open-book models perform better than all closed-book models. On average, the difference is more than one-and-a-half points on the evaluation scale. Even with an excluded GPT-4, the difference remains more than a full point on the evaluation scale.
- $T5_{BASE\_CLOSED}$  beats  $T5_{LARGE\_CLOSED}$  for closed-book QA: In contrast to the general NLP-benchmarks seen in 4.2.2,  $T5_{BASE\_CLOSED}$  actually beats  $T5_{LARGE\_CLOSED}$  for closed-book QA. Looking at the answers shows that  $T5_{LARGE\_CLOSED}$  has a very high rate of 0-point answers, suggesting a lack of understanding the question.
- Aleph Alphas luminous-base-control loses to  $T5_{BASE\_OPEN}$  and  $T5_{LARGE\_OPEN}$  for open-book QA: Since the advantage is only 0.1 and 0.2 respectively, it is likely that a different question set could have swapped this result. Nevertheless, the scores are close when combining all four categories.
- Aleph Alphas luminous-base-control is better at complicated "Explanations" but weaker at easy "Time or money-related questions":  $T5_{BASE\_OPEN}$  and  $T5_{LARGE\_OPEN}$  only score 2.1 and 2.5 for "Explanations" compared to luminous-base-controls 2.8. In contrast,  $T5_{BASE\_OPEN}$  and  $T5_{LARGE\_OPEN}$  handily beat luminous-base-control for "Time or money-related questions" with 3.0 and 2.5 compared to 2.0.

# 5

## Conclusion

The thesis wanted to find out if it was possible to train an economically feasible domain-specific LLMs by adopting a pretraining process. The resulting LLMs were compared to existing models to see if the models showed an improvement or similar results to existing larger models using RAG. This process contained the evaluation of different fine-tuning approaches. It also showed the associated costs of pretraining a model from scratch compared to adding additional training steps to an existing model. This chapter gives an overview of the findings for this thesis.

### 5.1. Using an available pretrained model is considerably cheaper than a model trained from scratch

The training results in 4.1.2 showed that pretraining is by far the largest driving cost factor. Even for the small  $T5_{BASE}$ , these costs easily exceed 1000\$. This is on par with other papers like GELECTRA, which had an estimated cost of about 1,300\$ [CSM20]. Due to the vanishing gradient issue, these costs would increase quickly for larger models. GELECTRA's larger 300 million parameter model already cost about 23,000\$ to train, despite still being a relatively small model. Comparing these costs to  $T5_{LARGE}$  shows that using a pretrained model is desirable for almost all use-cases. Close to no use-cases will be able to justify a cost increase of at least factor twelve. This effect is increased further since a pretrained model can easily be increased in size while keeping costs low.

The key cost factor for an external model is fine-tuning. While the total fine-tuning costs

for  $T5_{LARGE}$  were only slightly higher than the 35\$ for  $T5_{BASE}$ , this came with the downside of using only half of the number of epochs. Using the same amount of epochs would have doubled the costs. These extra costs didn't result in higher NLP benchmark scores, as shown in Chapter 4.2. Again, the higher number of layers (through additional transformer blocks) results in an increasing vanishing effect and, therefore, lower scores for the same training duration. It is therefore likely to be assumed that an increased training duration would have a stronger effect on  $T5_{LARGE}$  compared to  $T5_{BASE}$ , with better scores and a higher peak.

## 5.2. TPUs are cost-efficient compared to CPUs

Using an externally pretrained model would therefore focus more on the fine-tuning process. While these costs seem low at the moment, larger models could have increasingly higher costs to actually benefit from the parameter counts. As mentioned in Section 3.3.1, a CPU was used for fine-tuning. This was a result of the base premise of using an existing training setup and adapting it. With no working TPU fine-tuning setup available, no adaptation could be done.

The CPU costs 1.50\$ compared to the 8\$ for a TPUv3-8. But a CPU needs hours to finish a task TPUs could do in minutes. Assuming perfect efficiency, a single TPUv3-8 with its peak potential of 492 TFlops could process the open-book QA fine-tuning for  $T5_{LARGE}$  in less than three minutes. Even while assuming that perfect efficiency is very unlikely, this is a potential performance increase of up to 900 times, while being up to 168 times more cost-efficient. The potential lowest costs for fine-tuning  $T5_{LARGE}$  with a TPUv3-8 could therefore be 1,90\$. To enable large-scale fine-tuning with larger models, future work should rely on a TPU-based fine-tuning.

## 5.3. Closed-book QA doesn't show much promise

Domain-specific QA was the key goal for this thesis. The results showed that the approach used in this thesis didn't lead to any domain-specific closed-book results. The benchmarks shown in Section 4.2.2 were expected, as closed-book QA for generic questions is a very complicated topic, which even state-of-the-art LLMs struggle at. But even for domain-specific answers, these scores didn't get better. As Section 4.3.4 showed, on average,  $T5_{BASE\_CLOSED}$  had answers somewhere between obviously and non-obviously wrong. And even looking at specific question types, not a single average

was above 2.5, and therefore, on average, at least correct to some extent. For the seemingly more complicated "Explanations", average scores go as low as 1.0, indicating that at least half of the answers were either obviously wrong or didn't fit the answer at all.

$T5_{LARGE\_CLOSED}$  is a more interesting case. While the scores are even lower than  $T5_{BASE\_CLOSED}$  the distribution is completely different.  $T5_{BASE\_CLOSED}$  scores are more evenly distributed with a variance of only 1.3. In contrast,  $T5_{LARGE\_CLOSED}$  has a variance of 2.2, nearly twice as high. Most answers for  $T5_{LARGE\_CLOSED}$  are just 0-score answers, while the number of obvious or non-obvious wrong answers is very low. But the number of partly or fully right answers is again higher (but still relatively low, resulting in a very low average score). This shows that  $T5_{LARGE\_CLOSED}$  doesn't understand the question in most cases. More fine-tuning could likely improve this, which is interesting considering the fact that the benchmarks in Section 4.3 suggested some understanding of questions. In contrast, answers that  $T5_{LARGE\_CLOSED}$  understands have a good chance of being answered correctly.

Looking at the given answers reduces the potential viability of these models even further. Both  $T5_{CLOSED}$ -variants, but especially  $T5_{BASE\_CLOSED}$ , have learned to answer most cost-based answers with "kostenlos", which means "free of charge". In most cases, this is a non-obvious wrong answer, gaining a 2. And sometimes it's the correct answer just by pure chance, therefore gaining an undeserved 4. These answers show that using a different set of questions, for example, a set where no answer is "free of charge", would reduce the average scores even further.

For  $T5_{LARGE\_CLOSED}$ , the 0-score answers don't really contain any additional knowledge. But the 3- or 4-score answers show that the domain-specific documents affected the parameters of the model. Answers like in 1 are very likely a result of the domain-specific pretraining. For  $T5_{BASE\_CLOSED}$ , this doesn't show up in the answers, offering the possibility that either the domain-specific pretraining was not enough or fine-tuning erased these features. But less fine-tuning would probably result in the same 0-score issues as  $T5_{LARGE\_CLOSED}$ , which makes this option undesirable.

In total, this thesis found no evidence suggesting that a very small LLM like  $T5_{BASE\_CLOSED}$  could answer domain-specific closed-book questions with the employed training process. Due to the extremely low scores, it is also unlikely that more or less training could positively affect this result. While the scores for  $T5_{BASE\_CLOSED}$  were even lower, the actual answers showed more promise. It is very likely that more training, especially fine-tuning, would result in better scores. This also suggests that using an even larger model would have even more potential for closed-book QA. Nevertheless, since the average score for  $T5_{LARGE\_CLOSED}$  is still very low, it has to be increased by multiple steps to become viable. It therefore remains unlikely that training

a domain-specific LLM for closed-book QA with a relatively small size is possible.

## 5.4. Open-book QA might work

The same can't be said for open-book QA. While GPT-4 results are outstanding compared to any other result, this is expected due to the extremely high parameter count and training costs. But Aleph Alphas luminous-base-control already lost to  $T5_{BASE\_OPEN}$  and  $T5_{LARGE\_OPEN}$  on the total average score. Due to the low difference, a different question set could have resulted in different results, but it is likely to be assumed that all models would remain very close to each other. This is a notable achievement when comparing the parameter numbers.

It is interesting to note that luminous fails at seemingly easy questions. Especially short contexts that contain the answer and nothing else often result in a form of "No answer can be found". These questions require the least amount of logic and can often be answered with one exact line from the context. This is also very likely the reason why  $T5_{BASE\_OPEN}$  performs well on exactly these questions, exceeding even  $T5_{LARGE\_OPEN}$ .

This relationship turns around for more complicated answers. Just answering with a sentence from the context is often not enough to answer the question. Instead, more complicated logic has to be applied. Luminous-base-control exceeds  $T5_{BASE\_OPEN}$  by about 0.7 for "Explanations", going from on average wrong to on average right. For  $T5_{LARGE\_OPEN}$ , this difference goes down to 0.3. This might have multiple reasons. The larger parameter count likely comes into effect to enable a more complicated logic. This would also explain the difference between  $T5_{BASE\_OPEN}$  and  $T5_{LARGE\_OPEN}$ . But the later questions also have longer contexts, as noted in 4.3.2. While none exceed the maximum sequence length of  $T5_{BASE\_OPEN}$  or  $T5_{LARGE\_OPEN}$ , they might have issues dealing with such large contexts. And combining GPT-4 answers with the corresponding context exceeds the 512 maximum sequence length for some questions, showing that it can be a limiting factor.

But this issue doesn't come into effect since all answers from  $T5_{BASE\_OPEN}$  and  $T5_{LARGE\_OPEN}$  are very short. It is unlikely that this is due to the parameter size; instead, this might be a result of the available QA fine-tuning datasets. Both available German QA datasets focus on short answers, rarely exceeding two or three sentences. Instead of asking complicated "Explanation"-style questions, all questions focus on straight-forward questions. The expected answer is therefore always only a few sentences from the provided context. This is also the type of question most of the "Time or

money” and “Location or source”-related questions focus on.

The Deepset open-book approach also encourages bad sentence structure. The selected answers only contain the specific answer without any reference to the question. This leads to half-sentences lacking proper grammar and structure. In some cases, these might be complicated to understand, even while containing all relevant information (an example can be seen in 1).

When combining the previous aspects, it shows that using a larger domain-specific model with more training time might therefore yield better results for complicated questions. This oftentimes comes with a larger maximum sequence length as well, which might also be advantageous. And using different, more advanced datasets for fine-tuning on QA could have the largest effect. Combining these might give a better understanding if the domain-specific data during pretrained actual exceeds existing approaches.

### 5.5. Small domain-specific dataset

Another aspect that might have affected the results is the size of the domain-specific dataset. The administration dataset was relatively small, requiring a hundred duplications to achieve a notable size. T5<sub>LARGE</sub> had ten epochs using only these duplications, therefore increasing the effective size to a thousand duplications. And even with these duplications, training only took ten hours on the slower hardware. It is likely that this small size and training duration result in a small overall effect on the model parameters. But using less data would likely result in a model that lacks the general language capabilities to even understand the question. A larger domain-specific dataset might therefore increase the effect the domain-specific training has on the whole model.

### 5.6. Fine-tuning should focus on the required tasks

Small models have a tendency to overfit due to the low number of parameters. This was especially noticeable for classification, which resulted in high overfitting, even while having far less training time than open-book QA. But the results from 4.5 also showed that this over-fitting wasn't even enough to gain decent results for the classification test data set.

Multiple fine-tuning tasks are desirable for large LLMs to enable multiple use-cases. This doesn't transfer to small domain-specific LLMs. To enable different tasks, the



model needs to be larger to have enough parameters to be able to fulfil all tasks simultaneously. This larger model comes with the same costs as any larger model and therefore reduces the cost-efficiency for QA. Losing the ability to perform other tasks shouldn't be an issue if the focus is QA anyway. But if the focus is QA and additional other aspects, it becomes even more unlikely that a cost-efficient model can be trained. It is therefore advisable to use a different approach altogether for these use cases.

## 5.7. Possible future adaptation of this work

Combining these conclusions shows that closed-book QA leaves fewer options for further development. As described in Section 5.3, the main approach would be to use a larger model with far more fine-tuning. Using a larger domain or more duplications for pretraining might also be preferable, as shown in 5.5.

Open-book shows more promise and leaves an option for extension. As described in Section 5.4, using larger models could increase the quality of answers. Section 5.1 showed that using an existing pretrained model for this is strongly preferred. They are cheaper and show better results across the board. But as noted in 5.4, the quality of the available QA datasets might also have an effect. For German, the Deepset datasets are the only large-scale available datasets without a downside (other translated options exist but without translated answers). It is therefore advisable to use a different language, preferably English, for this task.

Using the English language could also enable the use of a different, larger domain. As highlighted in Section 5.5, this could increase the effect the domain-specific pretraining has on the whole model. Additionally, further fine-tuning with advanced QA-datasets could be conducted. To reduce costs, this fine-tuning should be run on a cost-efficient TPU. The results from this thesis suggest that this model would outperform larger models for open-book QA. Due to the extreme accuracy of GPT-4, it is unlikely to exceed it. But the cost per answer would be much cheaper.

But GPT-4 also had guaranteed access to the context containing the answer, since this thesis skipped the retrieval process for RAG. Including the retrieval process would likely result in worse answers for some questions due to suboptimal context selection. Comparing models with this increased complexity might yield interesting insights, where domain-specific pretraining shows an advantage.

## References

- [Ama93] Shun-ichi Amari. "Backpropagation and stochastic gradient descent method". In: *Neurocomputing* 5.4-5 (1993), pp. 185–196.
- [Gag94] Philip Gage. "A new algorithm for data compression". In: *C Users Journal* 12.2 (1994), pp. 23–38.
- [FJK96] Alan Frieze, Mark Jerrum, and Ravi Kannan. "Learning linear transformations". In: *Proceedings of 37th Conference on Foundations of Computer Science*. IEEE. 1996, pp. 359–368.
- [Hoc+01] Sepp Hochreiter et al. *Gradient flow in recurrent nets: the difficulty of learning long-term dependencies*. 2001.
- [HCL+03] Chih-Wei Hsu, Chih-Chung Chang, Chih-Jen Lin, et al. *A practical guide to support vector classification*. 2003.
- [AKO04] Mohamed Ibrahim Abouelhoda, Stefan Kurtz, and Enno Ohlebusch. "Replacing suffix trees with enhanced suffix arrays". In: *Journal of discrete algorithms* 2.1 (2004), pp. 53–86.
- [Chi04] Lin Chin-Yew. "Rouge: A package for automatic evaluation of summaries". In: *Proceedings of the Workshop on Text Summarization Branches Out, 2004*. 2004.
- [BN06] Christopher M Bishop and Nasser M Nasrabadi. *Pattern recognition and machine learning*. Vol. 4. 4. Springer, 2006.
- [Saz06] Murat Sazli. "A brief review of feed-forward neural networks". In: *Communications Faculty Of Science University of Ankara* 50 (2006-01), pp. 11–17. DOI: 10.1501/commua1-2\_0000000026.
- [TS10] Lisa Torrey and Jude Shavlik. "Transfer learning". In: *Handbook of research on machine learning applications and trends: algorithms, methods, and techniques*. IGI global, 2010, pp. 242–264.
- [BB12] James Bergstra and Yoshua Bengio. "Random search for hyper-parameter optimization." In: *Journal of machine learning research* 13.2 (2012).
- [HH12] Sarah Harris and David Harris. *Digital design and computer architecture*. 2nd. Morgan Kaufmann, 2012.

- [SN12] Mike Schuster and Kaisuke Nakajima. “Japanese and korean voice search”. In: *2012 IEEE international conference on acoustics, speech and signal processing (ICASSP)*. IEEE. 2012, pp. 5149–5152.
- [Gra13] Alex Graves. “Generating sequences with recurrent neural networks”. In: *arXiv preprint arXiv:1308.0850* (2013).
- [KB14] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014).
- [Bow+15] Samuel R Bowman et al. “Generating sentences from a continuous space”. In: *arXiv preprint arXiv:1511.06349* (2015).
- [Nie15] Michael A Nielsen. *Neural networks and deep learning*. Vol. 25. Determination press San Francisco, CA, USA, 2015.
- [Raj+16] Pranav Rajpurkar et al. “Squad: 100,000+ questions for machine comprehension of text”. In: *arXiv preprint arXiv:1606.05250* (2016).
- [Wu+16] Yonghui Wu et al. “Google’s neural machine translation system: Bridging the gap between human and machine translation”. In: *arXiv preprint arXiv:1609.08144* (2016).
- [Geh+17] Jonas Gehring et al. “Convolutional sequence to sequence learning”. In: *International conference on machine learning*. PMLR. 2017.
- [Vas+17] Ashish Vaswani et al. “Attention is All you Need”. In: *Advances in Neural Information Processing Systems*. Ed. by I. Guyon et al. Vol. 30. Curran Associates, Inc., 2017. URL: [https://proceedings.neurips.cc/paper\\_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf](https://proceedings.neurips.cc/paper_files/paper/2017/file/3f5ee243547dee91fbd053c1c4a845aa-Paper.pdf).
- [Dev+18] Jacob Devlin et al. “Bert: Pre-training of deep bidirectional transformers for language understanding”. In: *arXiv preprint arXiv:1810.04805* (2018).
- [Kud18] Taku Kudo. “Subword Regularization: Improving Neural Network Translation Models with Multiple Subword Candidates”. In: *CoRR* abs/1804.10959 (2018). arXiv: 1804.10959. URL: <http://arxiv.org/abs/1804.10959>.
- [KR18] Taku Kudo and John Richardson. “Sentencepiece: A simple and language independent subword tokenizer and detokenizer for neural text processing”. In: *arXiv preprint arXiv:1808.06226* (2018).
- [Rad+18] Alec Radford et al. “Improving language understanding by generative pre-training”. In: (2018).
- [Ris+18] Julian Risch et al. “Fine-Grained Classification of Offensive Language”. In: *Proceedings of GermEval 2018 (co-located with KONVENS)*. 2018-09, pp. 38–44.

- [SS18] Noam Shazeer and Mitchell Stern. “Adafactor: Adaptive learning rates with sublinear memory cost”. In: *International Conference on Machine Learning*. PMLR. 2018, pp. 4596–4604.
- [Che+19] Anthony Chen et al. “Evaluating question answering evaluation”. In: *Proceedings of the 2nd workshop on machine reading for question answering*. 2019, pp. 119–124.
- [Kal+19] Dhiraj Kalamkar et al. “A study of BFLOAT16 for deep learning training”. In: *arXiv preprint arXiv:1905.12322* (2019).
- [Kwi+19] Tom Kwiatkowski et al. “Natural questions: a benchmark for question answering research”. In: *Transactions of the Association for Computational Linguistics* 7 (2019), pp. 453–466.
- [Li+19] Ang Li et al. “A generalized framework for population based training”. In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 2019, pp. 1791–1799.
- [Raf+19] Colin Raffel et al. *Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer*. 2019. eprint: arXiv:1910.10683.
- [SSR19] Pedro Javier Ortiz Suárez, Benoît Sagot, and Laurent Romary. “Asynchronous pipeline for processing huge corpora on medium to low resource infrastructures”. In: *7th Workshop on the Challenges in the Management of Large Corpora (CMLC-7)*. Leibniz-Institut für Deutsche Sprache. 2019.
- [Wol+19] Thomas Wolf et al. *HuggingFace’s Transformers: State-of-the-art Natural Language Processing*. 2019. eprint: arXiv:1910.03771.
- [CSM20] Branden Chan, Stefan Schweter, and Timo Möller. “German’s next language model”. In: *arXiv preprint arXiv:2010.10906* (2020).
- [Jou+20] Norman P Jouppi et al. “A domain-specific supercomputer for training deep neural networks”. In: *Communications of the ACM* 63.7 (2020), pp. 67–78.
- [Kar+20] Vladimir Karpukhin et al. “Dense passage retrieval for open-domain question answering”. In: *arXiv preprint arXiv:2004.04906* (2020).
- [Lew+20] Patrick Lewis et al. “Retrieval-augmented generation for knowledge-intensive nlp tasks”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 9459–9474.
- [DeD21] DeDLOC. *Tokenizer training*. 2021. URL: <https://github.com/yandex-research/DeDLOC/tree/main/sahajbert/tokenizer> (visited on 2024-02-07).

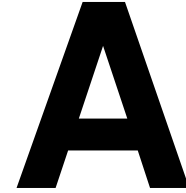
- [Del21] Julianna Delua. *Supervised vs. Unsupervised Learning: What's the Difference?* 2021. URL: <https://www.ibm.com/blog/supervised-vs-unsupervised-learning/> (visited on 2023-12-22).
- [Jou21] Norm Jouppi. *Google supercharges machine learning tasks with TPU custom chip*. 2021. URL: [https://github.com/google-research/text-to-text-transfer-transformer/blob/main/released\\_checkpoints.md#t511](https://github.com/google-research/text-to-text-transfer-transformer/blob/main/released_checkpoints.md#t511) (visited on 2024-02-13).
- [KiK21a] KiKaBeN. *Transformer's Encoder-Decoder*. 2021. URL: <https://kikaben.com/transformers-encoder-decoder/> (visited on 2023-12-07).
- [KiK21b] KiKaBeN. *Transformer's Encoder-Decoder*. 2021. URL: <https://kikaben.com/transformers-self-attention/> (visited on 2024-01-15).
- [MRP21] Timo Möller, Julian Risch, and Malte Pietsch. "GermanQuAD and GermanDPR: Improving non-English question answering and passage retrieval". In: *arXiv preprint arXiv:2104.12741* (2021).
- [Ope21] OpenAI. *DALL-E: Creating images from text*. 2021. URL: <https://openai.com/research/dall-e> (visited on 2023-12-07).
- [Raf+21] Colin Raffel et al. *Experimental T5 Pre-Trained Model Checkpoints*. 2021. URL: <https://cloud.google.com/blog/products/ai-machine-learning/google-supercharges-machine-learning-tasks-with-custom-chip> (visited on 2024-02-28).
- [Chu+22] Hyung Won Chung et al. "Scaling instruction-finetuned language models". In: *arXiv preprint arXiv:2210.11416* (2022).
- [Ger22] GermanT5. *GermanT5/t5-efficient-gc4-german-base-nl36 Tensorboard*. 2022. URL: <https://huggingface.co/GermanT5/t5-efficient-gc4-german-base-nl36/tensorboard> (visited on 2024-02-28).
- [Jai22] Shashank Mohan Jain. "Hugging face". In: *Introduction to Transformers for NLP: With the Hugging Face Library and Models to Solve Problems*. Springer, 2022, pp. 51–67.
- [Ope22] OpenAI. *Introducing ChatGPT*. 2022. URL: <https://openai.com/blog/chatgpt> (visited on 2024-01-31).
- [SN22] Gabriele Sarti and Malvina Nissim. "IT5: Large-scale Text-to-text Pretraining for Italian Language Understanding and Generation". In: *ArXiv preprint 2203.03759* (2022-03). URL: <https://arxiv.org/abs/2203.03759>.
- [Sch22a] Stefan Schweter. *Operative Config*. 2022. URL: [https://github.com/GermanT5/pre-training/blob/main/configurations/t5-efficient-small-el32/operative\\_config.gin](https://github.com/GermanT5/pre-training/blob/main/configurations/t5-efficient-small-el32/operative_config.gin) (visited on 2024-02-13).

- [Sch22b] Stefan Schweter. *Pre-Training of German T5 models*. 2022. URL: <https://github.com/GermanT5/pre-training> (visited on 2024-02-28).
- [Ach+23] Josh Achiam et al. “Gpt-4 technical report”. In: *arXiv preprint arXiv:2303.08774* (2023).
- [Bas23] Matthias Bastian. *GPT-4 has more than a trillion parameters - Report*. 2023. URL: <https://the-decoder.com/gpt-4-has-a-trillion-parameters/> (visited on 2024-01-31).
- [Chi+23] Wei-Lin Chiang et al. “Vicuna: An open-source chatbot impressing gpt-4 with 90%\* chatgpt quality”. In: *See https://vicuna.lmsys.org (accessed 14 April 2023)* (2023).
- [Cho+23] Aakanksha Chowdhery et al. “Palm: Scaling language modeling with pathways”. In: *Journal of Machine Learning Research* 24.240 (2023), pp. 1–113.
- [DT23] Matt Johnson Dougal Maclaurin David Duvenaud and Jamie Townsend. *Autograd*. 2023. URL: <https://github.com/HIPS/autograd> (visited on 2024-01-16).
- [Fla23] Flax. *The Flax philosophy*. 2023. URL: <https://flax.readthedocs.io/en/latest/philosophy.html> (visited on 2024-02-15).
- [Gan23] Lekhana Ganji. *One Hot Encoding in Machine Learning*. 2023. URL: <https://www.geeksforgeeks.org/ml-one-hot-encoding-of-datasets-in-python/> (visited on 2024-01-15).
- [HK23] Thomas F Heston and Charya Khun. “Prompt engineering in medical education”. In: *International Medical Education* 2.3 (2023), pp. 198–205.
- [Hum23] Michael Humor. *Understanding “tokens” and tokenization in large language models*. 2023. URL: <https://medium.com/@michaelhumor/understanding-tokens-and-tokenization-in-large-language-models-1058cd24b944> (visited on 2024-01-16).
- [Ken23] Patrick Kennedy. *Intel Announces 288 E-Core Sierra Forest Variant at Innovation 2023*. 2023. URL: <https://www.servethehome.com/intel-announces-288-e-core-sierra-forest-variant-at-innovation-2023/> (visited on 2024-02-13).
- [Kur23] M. Kurpicz-Briki. “Generative AI – was ist das und was kann sie bereits?” In: *SocietyByte* (2023). URL: [https://www.societybyte.swiss/2023/01/20/generative-ai-was-ist-das-und-was-kann-sie-bereits/#\\_ftn3](https://www.societybyte.swiss/2023/01/20/generative-ai-was-ist-das-und-was-kann-sie-bereits/#_ftn3) (visited on 2023-12-07).

- [Lub23] Mohammed Lubbad. *GPT-4 Parameters: Unlimited guide NLP's Game-Changer*. 2023. URL: <https://medium.com/@mlubbad/the-ultimate-guide-to-gpt-4-parameters-everything-you-need-to-know-about-nlps-game-changer-109b8767855a> (visited on 2024-01-31).
- [Sin+23] Karan Singhal et al. "Large language models encode clinical knowledge". In: *Nature* 620.7972 (2023), pp. 172–180.
- [Tou23] Toughdata. *Flan-T5-Quora-Question-Answering*. 2023. URL: [https://github.com/toughdata/Flan-T5-Quora-Question-Answering/blob/main/finetune\\_flan\\_t5.py](https://github.com/toughdata/Flan-T5-Quora-Question-Answering/blob/main/finetune_flan_t5.py) (visited on 2024-02-22).
- [Tou+23] Hugo Touvron et al. "Llama: Open and efficient foundation language models". In: *arXiv preprint arXiv:2302.13971* (2023).
- [Wu+23] Shijie Wu et al. "Bloomberggpt: A large language model for finance". In: *arXiv preprint arXiv:2303.17564* (2023).
- [Bad24] Land Baden-Württemberg. *Serviceportal Baden-Württemberg*. 2024. URL: <https://www.service-bw.de/zufi> (visited on 2024-02-16).
- [Cra24] Common Crawl. *Overview*. 2024. URL: <https://commoncrawl.org/overview> (visited on 2024-02-13).
- [Fac24a] Hugging Face. *Hugging Face*. 2024. URL: <https://huggingface.co/> (visited on 2024-02-14).
- [Fac24b] Hugging Face. *Language model training examples*. 2024. URL: <https://github.com/huggingface/transformers/tree/main/examples/flax/language-modeling> (visited on 2024-02-07).
- [Fac24c] Hugging Face. *Runtime evaluation*. 2024. URL: <https://github.com/huggingface/transformers/tree/main/examples/flax/language-modeling#runtime-evaluation> (visited on 2024-02-28).
- [Goo24a] Google. *Accelerate AI development with Google Cloud TPUs*. 2024. URL: <https://cloud.google.com/tpu> (visited on 2024-02-13).
- [Goo24b] Google. *Introduction to Cloud TPU*. 2024. URL: <https://cloud.google.com/tpu/docs/intro-to-tpu> (visited on 2024-02-15).
- [Goo24c] Google. *Regions and zones*. 2024. URL: <https://cloud.google.com/compute/docs/regions-zones> (visited on 2024-02-13).
- [Goo24d] Google. *TPU Research Cloud*. 2024. URL: <https://sites.research.google/trc/about/> (visited on 2024-01-16).
- [JAX24] JAX. *JAX: High-Performance Array Computing*. 2024. URL: <https://jax.readthedocs.io/en/latest/> (visited on 2024-02-07).

- [May24] Philip May. *GC4 Corpus*. 2024. URL: <https://german-nlp-group.github.io/projects/gc4-corpus.html> (visited on 2024-02-13).
- [Wu+24] Sean Wu et al. “Benchmarking Open-Source Large Language Models, GPT-4 and Claude 2 on Multiple-Choice Questions in Nephrology”. In: *NEJM AI* (2024), A1dbp2300092.





## Question Answer Pairs

### A.1. Time or money-related question

#### 1. Question: Was kostet es ein Auto anzumelden?

Context: Kraftfahrzeug Zulassung Kosten:\n Antrag: Je nach Verwaltungsaufwand ab EUR 2680 Kennzeichen: ab EUR 1020

- T5<sub>BASE.OPEN</sub>: ab eur 2680 kennzeichen: ab eur 1020
- T5<sub>BASE.CLOSED</sub>: gebührenfrei
- T5<sub>LARGE.OPEN</sub>: je nach Verwaltungsaufwand ab EUR 2680 Kennzeichen
- T5<sub>LARGE.CLOSED</sub>: ab dem 1. Januar 2019
- Luminous-base-control: Was kostet es, ein Auto anzumelden?
- GPT-4: Das Anmelden eines Autos kostet je nach Verwaltungsaufwand ab 26,80 EUR für den Antrag und zusätzlich ab 10,20 EUR für das Kennzeichen.

#### 2. Question: Wann ist mein Reisepass fertig?

Context: Reisepass - erstmalig oder nach Ablauf beantragen Bearbeitungsdauer:\nDie Bearbeitungsdauer beträgt mindestens drei Wochen. Bei verstärktem Bestelllaufkommen kann es produktionsbedingt zu einer Bearbeitungszeit von bis zu sechs Wochen kommen. Es wird empfohlen den Reisepass rechtzeitig vor Reiseantritt zu beantragen. Wenn Sie Ihren Reisepass schneller benötigen können Sie ihn auch im Expressverfahren beantragen. Geht der Express-Antrag rechtzeitig ein liegt Ihr Reisepass in der Regel am darauffolgenden dritten Werktag (es zählen Montag bis Freitag ohne Feiertage) in der Passbehörde abholbereit vor.\n

- T5<sub>BASE\_OPEN</sub>: in der regel am darauffolgenden dritten werktag (es zählen montag bis freitag ohne feiertage) in der passbehörde abholbereit
- T5<sub>BASE\_CLOSED</sub>: 14 tage nach der ausstellung
- T5<sub>LARGE\_OPEN</sub>: in der Regel am darauffolgenden dritten Werktag (es zählen Montag bis Freitag ohne Feiertage) in der Passbehörde abholbereit vor
- T5<sub>LARGE\_CLOSED</sub>: am nächsten Arbeitstag
- Luminous-base-control: Wird im Text erwähnt, wann der Reisepass fertig ist? Nein, es wird nicht erwähnt, wann der Reisepass fertig ist. Antwort: Es wird nicht erwähnt, wann der Reisepass fertig ist.
- GPT-4: Die Bearbeitungsdauer für deinen Reisepass beträgt mindestens drei Wochen. Bei verstärktem Bestellaufkommen kann es jedoch bis zu sechs Wochen dauern. Wenn du deinen Reisepass schneller benötigst, kannst du ihn im Expressverfahren beantragen. Dann liegt er in der Regel am darauffolgenden dritten Werktag in der Passbehörde abholbereit vor.

### 3. Question: Was kostet ein Reisepass?

Context: Reisepass - erstmalig oder nach Ablauf beantragen Kosten:\nWenn Sie unter 24 Jahre alt sind beträgt die Gebühr für einen Reisepass: EUR 3750 für einen Reisepass im Expressverfahren: EUR 6950 Ab dem 24. Lebensjahr beträgt die Gebühr für einen Reisepass: EUR 6000 für einen Reisepass im Expressverfahren: EUR 9200 Der Reisepass umfasst 32 Seiten. Sie können auch einen Reisepass mit 48 Seiten beantragen. In diesem Fall müssen Sie einen Zuschlag in Höhe von 22 Euro zahlen. Der Zuschlag beträgt 32 Euro für einen Reisepass im Expressverfahren. Die Gebühr (ohne Zuschläge) verdoppelt sich wenn: die Ausstellung außerhalb der behördlichen Dienstzeiten vorgenommen werden muss oder die Ausstellung nicht bei der örtlich zuständigen Passbehörde (Gemeinde der Hauptwohnung) beantragt wird. Wenn Sie einen Reisepass bei einer deutschen Botschaft oder konsularischen Vertretung z.B. bei Passverlust beantragen müssen Sie 21 Euro Zuschlag bezahlen.\n

- T5<sub>BASE\_OPEN</sub>: eur 3750 für einen reiseepass im expressverfahren: eur 9200
- T5<sub>BASE\_CLOSED</sub>: gebührenfrei
- T5<sub>LARGE\_OPEN</sub>: EUR 3750 für einen Reisepass im Expressverfahren: EUR 6950
- T5<sub>LARGE\_CLOSED</sub>: bis zu 24 Monaten
- Luminous-base-control: Wird ein Reisepass für einen unter 24-Jährigen ausgestellt, beträgt die Gebühr 3750 Euro.

- GPT-4: Die Kosten für einen Reisepass variieren je nach Alter und Schnelligkeit der Ausstellung. Wenn du unter 24 Jahre alt bist, kostet ein Reisepass 37,50 EUR und im Expressverfahren 69,50 EUR. Ab dem 24. Lebensjahr beträgt die Gebühr für einen Reisepass 60,00 EUR und im Expressverfahren 92,00 EUR. Für einen Reisepass mit 48 Seiten musst du einen Zuschlag von 22 Euro (oder 32 Euro im Expressverfahren) zahlen. Bedenke, dass sich die Gebühr verdoppelt, wenn die Ausstellung außerhalb der behördlichen Dienstzeiten erfolgt oder nicht bei der zuständigen Passbehörde beantragt wird. Bei einer Beantragung bei einer deutschen Botschaft oder konsularischen Vertretung, z.B. bei Passverlust, musst du einen Zuschlag von 21 Euro bezahlen.

4. Question: Wie lange dauert das Löschen meiner Personenbezogenen Daten?

Context: Personenbezogene Daten - Löschung beantragen Bearbeitungsdauer:\nDie Bearbeitungsdauer hängt von der Komplexität des Falles ab.\n

- $T5_{BASE\_OPEN}$ : 60 tagen
- $T5_{BASE\_CLOSED}$ : je nach auswahl des betroffenen sachverhalts (z. b. einwilligung, löschungsgrund, beschwerdegrund)
- $T5_{LARGE\_OPEN}$ : Bearbeitungsdauer: Die Bearbeitungsdauer hängt von der Komplexität des Falles ab.
- $T5_{LARGE\_CLOSED}$ : bis zu 30 Tagen
- Luminous-base-control: Wird im Text die Löschung von Personaldaten erwähnt? Ja, es wird erwähnt, dass die Bearbeitungsdauer von der Komplexität des Falles abhängt. Wird im Text auch die Dauer der Löschung erwähnt? Nein, es wird keine Dauer genannt. Antwort: Keine Antwort
- GPT-4: Die Bearbeitungsdauer für das Löschen deiner personenbezogenen Daten hängt von der Komplexität des Falles ab. Es gibt keine festgelegte Dauer.

5. Question: Bis wann muss ich meine Steuererklärung abgeben?

Context: Einkommensteuer - Erklärung abgeben Fristen:\nBei Pflichtveranlagung zur Einkommensteuer 2020: 1. November 2021 Bei Pflichtveranlagung zur Einkommensteuer 2021: 31. Oktober 2022 Sie können bei der zuständigen Stelle formlos beantragen dass diese Fristen verlängert werden. Wird Ihre Einkommensteuererklärung durch einen Angehörigen der steuerberatenden Berufe angefertigt gilt ab 2018 eine allgemein verlängerte Abgabefrist bis zum 28./29. Februar des

Zweitfolgejahres. Angesichts der durch die Corona-Pandemie verursachten Ausnahmesituation wird die Abgabefrist für das Jahr 2020 bei beratenen Steuerpflichtigen bis zum 31. August 2022 verlängert. Die Abgabefrist für die Einkommensteuererklärung 2021 wird bis zum 31. August 2023 verlängert. Bei Antragsveranlagung zur Einkommensteuer 2017: 31. Dezember 2021 Bei Antragsveranlagung zur Einkommensteuer 2018: 02. Januar 2023 Bei Antragsveranlagung zur Einkommensteuer 2019: 02. Januar 2024 Bei Antragsveranlagung zur Einkommensteuer 2020: 31. Dezember 2024 Bei Antragsveranlagung zur Einkommensteuer 2021: 31. Dezember 2025 Diese Fristen kann das Finanzamt nicht verlängern. Achtung: Wenn Sie Ihre Steuererklärung elektronisch aber nicht authentifiziert übermitteln geht sie beim Finanzamt erst mit Abgabe der von Ihnen unterschriebenen sogenannten komprimierten Steuererklärung ein. Nur die elektronische Übermittlung allein reicht in diesem Fall nicht aus. Beachten Sie das vor allem bei der Antragsveranlagung. Reichen Sie die komprimierte Steuererklärung erst nach Ablauf der vierjährigen Frist ein ist das zu spät.\n

- $T5_{BASE\_OPEN}$ : 31. dezember 2022
- $T5_{BASE\_CLOSED}$ : spätestens am 25. dezember 2018
- $T5_{LARGE\_OPEN}$ : bis zum 31. August 2022
- $T5_{LARGE\_CLOSED}$ : wenn Sie die Steuererklärung für das Jahr 2020 einreichen
- Luminous-base-control: Wird die Einkommensteuererklärung durch einen Angehörigen der steuerberatenden Berufe angefertigt gilt ab 2018 eine allgemein verlängerte Abgabefrist bis zum 28./29. Februar des Zweitfolgejahres. Angesichts der durch die Corona-Pandemie verursachten Ausnahmesituation wird die Abgabefrist für das Jahr 2020 bei beratenen Steuerpflichtigen bis zum 31. August 2022 verlängert.
- GPT-4: Die Fristen für die Abgabe der Einkommensteuererklärung variieren je nach Fall. Bei einer Pflichtveranlagung für das Jahr 2020 ist die Frist der 1. November 2021 und für das Jahr 2021 der 31. Oktober 2022. Wenn die Steuererklärung durch einen Angehörigen der steuerberatenden Berufe angefertigt wird, gilt eine verlängerte Abgabefrist bis zum 28./29. Februar des Zweitfolgejahres. Für das Jahr 2020 wurde die Frist auf den 31. August 2022 und für das Jahr 2021 auf den 31. August 2023 verlängert. Bei einer Antragsveranlagung variieren die Fristen zwischen dem 31. Dezember 2021 für das Jahr 2017 und dem 31. Dezember 2025 für das Jahr 2021. Bitte beachte, dass das Finanzamt diese Fristen nicht verlängern kann. Wenn

du deine Steuererklärung elektronisch übermittelst, muss auch eine unterschriebene komprimierte Steuererklärung eingereicht werden. Der Eingang dieser komprimierten Steuererklärung ist maßgeblich für die Fristberechnung.

6. Question: Was ist die Frist zum Anmelden einer Ehe?

Context: Eheschließung bei deutscher Staatsangehörigkeit anmelden Fristen:\nkeine\n

- $T5_{BASE\_OPEN}$ : 30 tagen
- $T5_{BASE\_CLOSED}$ : 31. dezember 2004
- $T5_{LARGE\_OPEN}$ : Frist von sechs Wochen
- $T5_{LARGE\_CLOSED}$ : die Frist, in der sich die Anmeldung beim Standesamt oder bei der zuständigen Behörde des Wohnortes des Antragstellers erübrigt
- Luminous-base-control: Wird im Text eine Frist für die Anmeldung einer Ehe erwähnt? Nein, es wird keine Frist genannt. Antwort: Keine.
- GPT-4: Es gibt keine festgelegte Frist zum Anmelden einer Ehe bei deutscher Staatsangehörigkeit.

7. Question: Was kostet eine Meldebescheinigung?

Context: Meldebescheinigung beantragen Kosten:\nDie Höhe der Gebühr richtet sich nach der örtlichen Gebührensatzung. Die elektronische Meldebescheinigung wird unentgeltlich erteilt. Für Sozialleistungen wie zum Beispiel Krankenversicherung Rentenversicherung Kindergeld Wohngeld ist die Meldebescheinigung kostenfrei.\n

- $T5_{BASE\_OPEN}$ : die gebühr richtet sich nach der örtlichen gebührensatzung
- $T5_{BASE\_CLOSED}$ : gebührenfrei
- $T5_{LARGE\_OPEN}$ : kostenlos
- $T5_{LARGE\_CLOSED}$ : kostenlos
- Luminous-base-control: Wird im Text keine Antwort gegeben, kann ich keine Antwort geben.
- GPT-4: Die Kosten für eine Meldebescheinigung richten sich nach der örtlichen Gebührensatzung. Eine elektronische Meldebescheinigung sowie eine Meldebescheinigung für Sozialleistungen wie Krankenversicherung, Rentenversicherung, Kindergeld oder Wohngeld wird kostenfrei erteilt.

8. Question: Was kostet es meinen Hund anzumelden?

Context: Hundesteuer - Hund anmelden Kosten:\n-\n

- $T5_{BASE\_OPEN}$ : hundesteuer - hund anmelden kosten: -
- $T5_{BASE\_CLOSED}$ : gebührenfrei
- $T5_{LARGE\_OPEN}$ : Hundesteuer
- $T5_{LARGE\_CLOSED}$ : kostenlos
- Luminous-base-control: Wird der Hund in der Steuererklärung des Hundes erwähnt? Nein, es wird nicht erwähnt, was es kostet, den Hund anzumelden. Antwort: Ich konnte keine Antwort finden
- GPT-4: Der Kontext gibt keine spezifischen Kosten für die Anmeldung eines Hundes an. Diese könnten je nach Ort variieren und sind daher vermutlich in der örtlichen Gebührensatzung festgelegt.

9. Question: Wie viel kostet eine Geburtsurkunde?

Context: Geburtsurkunde beantragen Kosten:\nGeburtsurkunde oder beglaubigter Ausdruck aus dem Geburtsregister: je EUR 1200 Internationale Geburtsurkunde: EUR 1200\n

- $T5_{BASE\_OPEN}$ : je eur 1200
- $T5_{BASE\_CLOSED}$ : 50 euro pro geburt
- $T5_{LARGE\_OPEN}$ : je EUR 1200
- $T5_{LARGE\_CLOSED}$ : kostet pro Kind 1 Euro
- Luminous-base-control: Wird im Text eine Geburtsurkunde erwähnt? Ja, es wird eine Geburtsurkunde erwähnt. Wird im Text der Preis für eine Geburtsurkunde erwähnt? Ja, es wird gesagt, dass sie 1200 Euro kostet. Antwort: Die Geburtsurkunde kostet 1200 Euro.
- GPT-4: Eine Geburtsurkunde oder ein beglaubigter Ausdruck aus dem Geburtsregister kostet je 12,00 EUR. Eine internationale Geburtsurkunde kostet ebenfalls 12,00 EUR.

## A.2. Location or source

1. Question: Wo kann ich ein Auto anmelden?

Context: Kraftfahrzeug Zulassung Zuständige Stelle:\nÖrtlich zuständige Zulassungsbehörden <https://www.strassenverkehrsamt.de/kfz-zulassungsstelle>

- $T5_{BASE\_OPEN}$ : bei der örtlich zuständigen zulassungsbehörden <https://www.strassenverkehrsamt.de/kfz-zulassungsstelle>

- $T5_{BASE\_CLOSED}$ : auf einer internetplattform
- $T5_{LARGE\_OPEN}$ : auf der Internetseite des Strassenverkehrsamt.de/kfz-zulassungsstelle
- $T5_{LARGE\_CLOSED}$ : in Deutschland, Österreich und der Schweiz
- Luminous-base-control: Antwort: Sie können ein Auto bei einer örtlichen Zulassungsstelle anmelden.
- GPT-4: Ein Auto kannst du bei der örtlich zuständigen Zulassungsbehörde anmelden. Informationen zu den Zulassungsstellen findest du unter folgendem Link: <https://www.strassenverkehrsamt.de/kfz-zulassungsstelle>

2. Question: Wo melde ich mich für die Infektionsschutzbelehrung an?

Context: Infektionsschutzbelehrung inklusive Bescheinigung beantragen Zuständige Stelle:\ndas Gesundheitsamt Gesundheitsamt ist wenn Sie in Stuttgart Mannheim oder Heilbronn wohnen: die jeweilige Stadtverwaltung ansonsten das Landratsamt.

- $T5_{BASE\_OPEN}$ : das gesundheitsamt gesundheitsamt
- $T5_{BASE\_CLOSED}$ : in der regel durch eine ausweise bei der zuständigen behörde oder durch ein gesundheitsamt
- $T5_{LARGE\_OPEN}$ : das Landratsamt
- $T5_{LARGE\_CLOSED}$ : das Gesundheitsamt
- Luminous-base-control: Wird im Text die Adresse des Gesundheitsamtes genannt? Ja, es wird das Gesundheitsamt als zuständige Stelle genannt. Wird im Text auch die Adresse des Landratsamtes genannt? Nein, es wird nur das Gesundheitsamt genannt. Antwort: Die Adresse des Gesundheitsamtes ist die zuständige Stelle für die Infektionsschutzbelehrung.
- GPT-4: Für die Anmeldung zur Infektionsschutzbelehrung inklusive Bescheinigung ist das Gesundheitsamt zuständig. Wenn du in Stuttgart, Mannheim oder Heilbronn wohnst, ist die jeweilige Stadtverwaltung zuständig, ansonsten das Landratsamt.

3. Question: Wo stell ich einen Strafantrag?

Context: Strafantrag stellen Zuständige Stelle:\ndie Polizei Staatsanwaltschaft oder das Amtsgericht\n

- $T5_{BASE\_OPEN}$ : die polizei staatsanwaltschaft oder das amtsgericht
- $T5_{BASE\_CLOSED}$ : in der regel durch eine strafuntersuchung
- $T5_{LARGE\_OPEN}$ : die Polizei Staatsanwaltschaft oder das Amtsgericht

- T5<sub>LARGE\_CLOSED</sub>: in der Regel bei der zuständigen Behörde
- Luminous-base-control: Wird im Text erwähnt, wo man einen Strafantrag stellen kann? Ja, es wird erwähnt, dass die Polizei, die Staatsanwaltschaft oder das Amtsgericht die zuständigen Stellen sind, die einen Strafantrag stellen können. Antwort: Die zuständigen Stellen, die einen Strafantrag stellen können, sind die Polizei, die Staatsanwaltschaft oder das Amtsgericht.
- GPT-4: Einen Strafantrag kannst du bei der Polizei, der Staatsanwaltschaft oder dem Amtsgericht stellen.

#### 4. Question: Woher bekomme ich Wohngeld?

Context: Wohngeld beantragen Zuständige Stelle:\nDie Wohngeldbehörde. Die Wohngeldbehörde ist je nach Wohnort die Stadtverwaltung oder das Landratsamt. Hinweis: Für die Einwohnerinnen und Einwohner einzelner Großer Kreisstädte ist möglicherweise das Landratsamt zuständig. Die Verwaltungen der Gemeinden die keine Wohngeldzuständigkeit haben nehmen Ihren Antrag aber entgegen und leiten ihn an die zuständige Stelle weiter.\n

- T5<sub>BASE\_OPEN</sub>: wohngeld beantragen zuständige stelle: die wohngeldbehörde ist je nach wohnort die stadtverwaltung oder das landratsamt
- T5<sub>BASE\_CLOSED</sub>: vom amt für wohnen und soziales
- T5<sub>LARGE\_OPEN</sub>: je nach Wohnort die Stadtverwaltung oder das Landratsamt
- T5<sub>LARGE\_CLOSED</sub>: für die kreisfreien Städte und die kreisfreien Städte
- Luminous-base-control: Wird im Text erwähnt, wo man Wohngeld bekommt? Ja, es wird erwähnt, dass die Wohngeldbehörde zuständig ist. Wird im Text auch angegeben, wer für die Einwohnerinnen und Einwohner einzelner Großer Kreisstädte zuständig ist? Ja, es wird erwähnt, dass das Landratsamt für diese zuständig ist. Antwort: Die Wohngeldbehörde ist für die Einwohnerinnen und Einwohner einzelner Großer Kreisstädte zuständig. Das Landratsamt ist für die Einwohnerinnen und Einwohner der einzelnen Großer Kreisstädte zuständig.
- GPT-4: Wohngeld kannst du bei der Wohngeldbehörde beantragen. Je nach Wohnort ist dies die Stadtverwaltung oder das Landratsamt. Auch wenn deine Gemeindeverwaltung nicht die Zuständigkeit für Wohngeld hat, können sie deinen Antrag entgegennehmen und an die zuständige Stelle weiterleiten.

#### 5. Question: Wo erhalte ich eine Meldebescheinigung?



Context: Meldebescheinigung beantragen Zuständige Stelle:\ndie Meldebehörde Ihres Wohnorts Meldebehörde ist die Gemeinde-/Stadtverwaltung Ihres Wohnortes oder die Verwaltungsgemeinschaft oder die Gemeinde die die Aufgaben der Meldebehörde für Ihre Wohnortgemeinde erfüllt.\n

- $T5_{BASE\_OPEN}$ : meldebescheinigung
- $T5_{BASE\_CLOSED}$ : in der regel bei der meldebehörde (landratsamt)
- $T5_{LARGE\_OPEN}$ : die Gemeinde-/Stadtverwaltung Ihres Wohnortes oder die Verwaltungsgemeinschaft oder die Gemeinde
- $T5_{LARGE\_CLOSED}$ : die Meldebehörde
- Luminous-base-control: Wird im Text eine Meldebescheinigung erwähnt? Nein, es wird nicht erwähnt, wo man eine Meldebescheinigung erhalten kann. Antwort: Keine Antwort
- GPT-4: Eine Meldebescheinigung erhältst du bei der Meldebehörde deines Wohnorts. Das kann die Gemeinde-/Stadtverwaltung deines Wohnortes sein oder die Verwaltungsgemeinschaft oder die Gemeinde, die die Aufgaben der Meldebehörde für deine Wohnortgemeinde erfüllt.

#### 6. Question: Was ist die Zuständige Hilfe für Blinderhilfe?

Context: Blinderhilfe beantragen Zuständige Stelle:\ndas Sozialamt Sozialamt ist wenn Sie in einem Stadtkreis wohnen: die Stadtverwaltung wenn Sie in einem Landkreis wohnen: das Landratsamt Hinweis: Teilweise haben die Landkreise die Zuständigkeit auf größere Städte in ihrem Kreisgebiet übertragen. Wohnen Sie in einem Landkreis kann Ihnen das Landratsamt oder die Gemeindeverwaltung Ihres Wohnortes die zuständige Behörde nennen.\n

- $T5_{BASE\_OPEN}$ : das sozialamt sozialamt
- $T5_{BASE\_CLOSED}$ : die zuständige anlaufstelle für blinderhilfe in deutschland
- $T5_{LARGE\_OPEN}$ : das Sozialamt Sozialamt
- $T5_{LARGE\_CLOSED}$ : die Stadtverwaltung
- Luminous-base-control: Wird im Text die "zuständige Behörde" für die Blinderhilfe erwähnt? Ja, es wird die "zuständige Behörde" genannt. Wird im Text die Zuständigkeit für die Blinderhilfe erwähnt? Ja, es wird erwähnt, dass die Zuständigkeit für die Blinderhilfe von der Stadt oder dem Landkreis übernommen werden kann. Antwort: Die zuständige Behörde für die Blinderhilfe ist das Sozialamt.

- GPT-4: Die zuständige Stelle für die Beantragung von Blindenhilfe ist das Sozialamt. Wenn Sie in einem Stadtkreis wohnen, ist die Stadtverwaltung zuständig, und wenn Sie in einem Landkreis wohnen, ist es das Landratsamt. In einigen Fällen haben die Landkreise die Zuständigkeit auf größere Städte in ihrem Kreisgebiet übertragen. Sollten Sie in einem Landkreis wohnen, können Ihnen das Landratsamt oder die Gemeindeverwaltung Ihres Wohnortes die zuständige Behörde nennen.

### A.3. Yes/No-Questions

#### 1. Question: Muss ich einen Hund anmelden?

Context: Hundesteuer - Hund anmelden Wenn Sie einen Hund halten müssen Sie ihn anmelden und Hundesteuer bezahlen. Die Hundesteuer ist eine Gemeindesteuer. Jede Gemeinde kann dafür eigene Regelungen festlegen. Daher können die Voraussetzungen und die Höhe der Steuer von Gemeinde zu Gemeinde unterschiedlich sein.\n

- T5<sub>BASE\_OPEN</sub>: muss er angemeldet werden und hundesteuer bezahlen
- T5<sub>BASE\_CLOSED</sub>: zum einen beim zuständigen gewerbeamt oder bei der zuständigen gemeindeverwaltung oder beim zuständigen amtsgericht
- T5<sub>LARGE\_OPEN</sub>: müssen Sie ihn anmelden und Hundesteuer bezahlen
- T5<sub>LARGE\_CLOSED</sub>: Die Gemeinde erhebt eine jährliche Gebühr und richtet sich nach der Höhe des Hundesteuersatzes
- Luminous-base-control: Ja, Sie müssen einen Hund anmelden.
- GPT-4: Ja, wenn du einen Hund hältst, musst du ihn anmelden und Hundesteuer bezahlen. Die spezifischen Regelungen und die Höhe der Steuer können jedoch von Gemeinde zu Gemeinde variieren.

#### 2. Question: Muss ich die Adresse im Personalausweis wirklich ändern lassen?

Context: Personalausweis - Adresse ändern lassen Hinweise:\nLassen Sie Ihren Personalausweis nicht ändern begehen Sie eine Pflichtverletzung. Hier finden Sie umfassende Informationen zum Personalausweis und wie Sie ihn beantragen.\n

- T5<sub>BASE\_OPEN</sub>: müssen sie ihren personalausweis nicht ändern lassen
- T5<sub>BASE\_CLOSED</sub>: muss ich zuvor die adresse des personalausweises angeben
- T5<sub>LARGE\_OPEN</sub>: Lassen Sie Ihren Personalausweis nicht ändern

- T5<sub>LARGE\_CLOSED</sub>: Der Personalausweis ist ein amtliches Dokument, das den Namen des Antragstellers und die Identifikationsnummer enthält
- Luminous-base-control: Ja, Sie müssen die Adresse im Personalausweis wirklich ändern lassen.
- GPT-4: Ja, es ist eine Pflicht, die Adresse im Personalausweis ändern zu lassen, wenn sich dein Wohnort ändert. Wenn du deinen Personalausweis nicht änderst, begehest du eine Pflichtverletzung.

### 3. Question: Ist das Anmelden eines Autos kostenlos?

Context: Kraftfahrzeug Zulassung Kosten:\nAntrag: Je nach Verwaltungsaufwand ab EUR 2680 Kennzeichen: ab EUR 1047

- T5<sub>BASE\_OPEN</sub>: je nach verwaltungsaufwand ab eur 2680 kennzeichen: ab eur 1047
- T5<sub>BASE\_CLOSED</sub>: kostenlos
- T5<sub>LARGE\_OPEN</sub>: bis EUR 1047 Kennzeichen
- T5<sub>LARGE\_CLOSED</sub>: bis zu 5
- Luminous-base-control: Ja, das Anmelden eines Autos ist kostenlos.
- GPT-4: Das Anmelden eines Autos ist nicht kostenlos. Je nach Verwaltungsaufwand beginnen die Kosten für den Antrag bei 26,80 EUR und die Kosten für das Kennzeichen bei 10,47 EUR.

### 4. Question: Benötige ich einen Personalausweis für die Anmeldung zur Eheschließung?

Context: Eheschließung bei deutscher Staatsangehörigkeit anmelden Erforderliche Unterlagen:\nngültiger Personalausweis oder Reisepass bei Geburt im Ausland: Geburtsurkunde bei Geburt im Inland: beglaubigter Ausdruck aus dem Geburtenregister mit Hinweisteil oder beglaubigte Abschrift aus dem Geburtenbuch mit Hinweisteil erweiterte Meldebescheinigung der Meldebehörde: Sie darf nicht älter als vier Wochen sein. Sie ist nicht zu verwechseln mit aufenthaltsrechtlichen Erlaubnissen für ausländische Staatsangehörige. In manchen Gemeinden kann das Standesamt die erweiterte Meldebescheinigung für Sie ausdrucken. Eine einfache Meldebescheinigung genügt nicht. Die zuständige Stelle kann weitere Unterlagen verlangen zum Beispiel die Einbürgerungsurkunde. Hinweis: Der beglaubigte Ausdruck aus dem Geburtenregister enthält außer den Angaben zur Geburt (einschließlich Geburtszeit und Angaben zu den Eltern) auch spätere Änderungen wie etwa durch Adoption oder Namensänderung. Hinweise werden nur auf Verlangen aufgenommen.\n

- T5<sub>BASE\_OPEN</sub>: benötigter personalausweis oder reiseepass bei geburt im ausland: geburtsurkunde bei geburt im inland: beglaubigter ausdruck aus dem geburtenregister mit hinweisteil oder beglaubigte abschrift aus dem geburtenbuch mit hinweisteil
- T5<sub>BASE\_CLOSED</sub>: benötigt der ehemann einen amtlichen ausweis, den er nach der anmeldung bei der zuständigen behörde erhält
- T5<sub>LARGE\_OPEN</sub>: benötigter personalausweis oder reiseepass bei geburt im ausland: geburtsurkundeein gültiger Personalausweis oder Reisepass bei Geburt im Ausland: beglaubigter Ausdruck aus dem Geburtenregister mit Hinweisteil oder beglaubigte Abschrift aus dem Geburtenbuch mit Hinweis- teil bei geburt im inland: beglaubigter ausdruck aus dem geburtenregister mit hinweisteil oder beglaubigte abschrift aus dem geburtenbuch mit hinweisteil
- T5<sub>LARGE\_CLOSED</sub>: die Geburtsurkunde
- Luminous-base-control: Nein, Sie benötigen keinen Personalausweis für die Anmeldung zur Eheschließung.
- GPT-4: Ja, für die Anmeldung zur Eheschließung benötigst du einen gültigen Personalausweis oder Reisepass. Weitere erforderliche Unterlagen können eine Geburtsurkunde oder einen beglaubigten Ausdruck aus dem Geburten- register sowie eine erweiterte Meldebescheinigung der Meldebehörde sein, die nicht älter als vier Wochen sein darf. Je nach Situation können noch weitere Unterlagen verlangt werden.

5. Question: Ist das Anfordern eines Urteils kostenlos?

Context: Urteil anfordern Kosten:\nfür eine Recherche in der baden-württembergischen Rechtsprechungsdatenbank für den privaten Gebrauch: keine für Entschei- dungen die Sie bei baden-württembergischen Gerichten anfordern: EUR 1600 pro Entscheidung Hinweis: Gerichte außerhalb von Baden-Württemberg können andere Gebühren erheben.\n

- T5<sub>BASE\_OPEN</sub>: für eine recherche in der baden-württembergischen recht- sprechungsdatenbank für den privaten gebrauch
- T5<sub>BASE\_CLOSED</sub>: kostenlos
- T5<sub>LARGE\_OPEN</sub>: für den privaten Gebrauch
- T5<sub>LARGE\_CLOSED</sub>: für die Rechtsprechung
- Luminous-base-control: Ja, das Anfordern eines Urteils ist kostenlos.
- GPT-4: Das Anfordern eines Urteils ist nicht generell kostenlos. Eine Recherche in der baden-württembergischen Rechtsprechungsdatenbank für den privaten

Gebrauch ist kostenlos. Wenn du jedoch Entscheidungen bei baden-württembergischen Gerichten anforderst, kostet jede Entscheidung 16,00 EUR. Beachte, dass Gerichte außerhalb von Baden-Württemberg andere Gebühren erheben können.

6. Question: Kann ich Abfälle aus Renovierungen als Sperrmüll anmelden?

Context: Sperrmüll entsorgen Voraussetzungen:\nBei Ihrem Abfall handelt es sich um Sperrmüll. Auskunft darüber welche Gegenstände Sie in Ihrem Stadt-/Landkreis als Sperrmüll entsorgen dürfen erhalten Sie bei dem für Sie zuständigen Abfallwirtschaftsbetrieb. Beispiele für Sperrmüll: Betten Matratzen Polstermöbel Schrankteile Tische Stühle Spiegel Koffer Teppiche Gartenmöbel Achtung: Abfälle aus Renovierungen oder Umbau wie alte Fliesen oder Sanitärkeramik gehören nicht zum Sperrmüll. Diese müssen Sie als Bauschutt getrennt entsorgen. Tape-tenabfälle müssen Sie als Restabfall entsorgen.\n

- T5<sub>BASE\_OPEN</sub>: bei dem für sie zuständigen abfallwirtschaftsbetrieb
- T5<sub>BASE\_CLOSED</sub>: sowohl als bauabfälle als auch als bauabfälle
- T5<sub>LARGE\_OPEN</sub>: müssen Sie als Bauschutt getrennt entsorgen
- T5<sub>LARGE\_CLOSED</sub>: "Altpapier", "Altpapiercontainer", "Altpapiercontainer", "Altpapiercontainer", "Altpapiercontainer"
- Luminous-base-control: Nein, Abfälle aus Renovierungen oder Umbauten gehören nicht zum Sperrmüll.
- GPT-4: Nein, du kannst Abfälle aus Renovierungen nicht als Sperrmüll anmelden. Diese gehören nicht zum Sperrmüll und müssen als Bauschutt getrennt entsorgt werden. Tapetenabfälle müssen als Restabfall entsorgt werden.

## A.4. Explanations

1. Question: Wie kann ich mein Kind im Kindergarten anmelden?

Context: Kindertageseinrichtung - Kind für Aufnahme anmelden Verfahrensablauf:\nNehmen Sie Kontakt zur Leitung der gewünschten Einrichtung auf. Fragen Sie nach den Öffnungszeiten Mahlzeitenangebot und nach den erzieherischen Schwerpunkten. Klären Sie vor Ort das genaue Anmeldeverfahren und die Anmeldefristen. Erkundigen Sie sich ob Sie Ihr Kind für mehrere Einrichtungen anmelden können. Von der Einrichtung für die Sie sich entscheiden oder dessen Träger erhalten

Sie ein Formular für die ärztliche Untersuchung ausgehändigt. Lassen Sie es von der untersuchenden Ärztin oder dem untersuchenden Arzt ausfüllen. Die ärztliche Bescheinigung sollte Ihnen bestätigen dass gegen den Besuch der Einrichtung keine Bedenken bestehen. Vor der Aufnahme in eine Einrichtung schließt der Träger mit Ihnen einen Betreuungsvertrag ab. Er enthält rechtliche Vereinbarungen beispielsweise über die tägliche Betreuungszeit Betreuungskosten und Kündigungsfristen. Hinweis: Bei Kindern mit Behinderung empfiehlt es sich vor der Aufnahme mit der Einrichtung und seinem Träger Kontakt aufzunehmen um eventuell erforderliche zusätzliche Unterstützung zu klären.\n

- T5<sub>BASE\_OPEN</sub>: für aufnahme anmelden
- T5<sub>BASE\_CLOSED</sub>: hierfür ist eine anmeldung beim zuständigen jugendamt erforderlich
- T5<sub>LARGE\_OPEN</sub>: Sie können vor Ort das genaue Anmeldeverfahren und die Anmeldefristen beantragen
- T5<sub>LARGE\_CLOSED</sub>: die Aufnahmevoraussetzungen sowie die Aufnahmevoraussetzungen
- Luminous-base-control: Wird im Text die Anmeldung eines Kindes im Kindergarten erwähnt? Ja, es wird eine Anmeldung erwähnt. Wird im Text das Verfahren zur Anmeldung eines Kindes im Kindergarten beschrieben? Ja, es wird beschrieben, wie man sich an die Einrichtung wendet, um sich nach den Öffnungszeiten, dem Essensangebot und den pädagogischen Schwerpunkten zu erkundigen. Wird im Text das Anmeldeverfahren und die Anmeldefristen beschrieben? Ja, es wird beschrieben, wie man sich vor Ort über das Anmeldeverfahren und die Anmeldefristen informiert. Wird im Text erwähnt, dass man sein Kind für mehrere Einrichtungen anmelden kann? Ja, es wird erwähnt, dass man sein Kind für mehrere Einrichtungen anmelden kann. Wird im Text ein Formular für die ärztliche Untersuchung erwähnt? Ja, es wird erwähnt, dass man ein Formular für die ärztliche Untersuchung erhält, das ausgefüllt werden muss. Antwort: Um Ihr Kind im Kindergarten anzumelden, wenden Sie sich an die Einrichtung und erkundigen Sie sich nach den Öffnungszeiten, dem Essensangebot und den pädagogischen Schwerpunkten. Sie können sich vor Ort über das Anmeldeverfahren und die Anmeldefristen informieren. Sie können Ihr Kind für mehrere Einrichtungen anmelden. Sie erhalten ein Formular für die ärztliche Untersuchung, das ausgefüllt werden muss.
- GPT-4: Um dein Kind im Kindergarten anzumelden, solltest du zuerst Kontakt zur Leitung der gewünschten Einrichtung aufnehmen. Du solltest dich

dort über die Öffnungszeiten, Mahlzeitenangebote und erzieherischen Schwerpunkte informieren. Kläre vor Ort das genaue Anmeldeverfahren und die Anmeldefristen. Erkundige dich, ob du dein Kind für mehrere Einrichtungen anmelden kannst. Von der Einrichtung, für die du dich entscheidest, oder von deren Träger erhältst du ein Formular für die ärztliche Untersuchung, das du von der Ärztin oder dem Arzt ausfüllen lässt. Vor der Aufnahme schließt der Träger mit dir einen Betreuungsvertrag ab, der rechtliche Vereinbarungen wie Betreuungszeit, Betreuungskosten und Kündigungsfristen enthält.

## 2. Question: Wie bekomme ich einen Reisepass?

Context: Reisepass - erstmalig oder nach Ablauf beantragen Verfahrensablauf:\nSie müssen Ihren Reisepass persönlich beantragen und die erforderlichen Unterlagen vorlegen. Der Antrag wird vor Ort in der Passbehörde aufgenommen. Sie müssen nur unterschreiben und die Erklärung zur deutschen Staatsangehörigkeit ausfüllen. Eine Unterschrift ist bei Kindern ab 10 Jahren erforderlich. Für die Antragstellung ist die Abgabe von Fingerabdrücken gesetzlich verpflichtend (flacher Abdruck des linken und des rechten Zeigefingers). Die Abgabe von Fingerabdrücken ist bei Kindern ab 6 Jahren erforderlich. Hinweis: Bei Fehlen eines Zeigefingers ungenügender Qualität des Fingerabdrucks oder Verletzungen der Fingerkuppe wird ersatzweise ein anderer Abdruck genommen. Fingerabdrücke werden nur dann nicht abgenommen wenn dies aus medizinischen dauerhaft bestehenden Gründen unmöglich ist. Ab Antragstellung dauert es in der Regel mindestens drei Wochen bis Sie Ihren Reisepass in der Passbehörde abholen können. Wenn Sie Ihren Reisepass schneller benötigen können Sie ihn auch im Expressverfahren beantragen. Dafür wird zusätzlich ein Zuschlag zur Gebühr erhoben. Der Reisepass wird vom Passproduzenten der Bundesdruckerei GmbH nach der Herstellung an die antragstellende Passbehörde verschickt. Je nach Gemeinde werden Sie benachrichtigt sobald Sie Ihren Reisepass abholen können. Die Benachrichtigungsinformation der Passbehörde enthält meistens auch einen Vordruck der Abholvollmacht. Somit können Sie Ihren Reisepass selbst abholen oder ihn von einer bevollmächtigten Person abholen lassen. Die bevollmächtigte Person muss sich gegenüber der Passbehörde vorher ausweisen und die Abholvollmacht vorlegen. Sie müssen Ihren alten Reisepass beim Empfang des neuen Reisepasses abgeben. Auf Wunsch können Sie Ihren alten Reisepass entwertet als Andenken wieder mitnehmen.\n

- $T5_{BASE\_OPEN}$ : wie bekomme ich meinen reisepass persönlich
- $T5_{BASE\_CLOSED}$ : mit der post („ausweis“)





[illegible]

- T5<sub>LARGE\_CLOSED</sub>: im Jahr 2017 nur für die ersten beiden Lebensjahre
- Luminous-base-control: Wird im Text erwähnt, wann man Elterngeld bekommt?  
Ja, es wird erwähnt, dass man es erst nach der Geburt des Kindes bekommt.
- GPT-4: Elterngeld erhältst du unter bestimmten Voraussetzungen: Du musst in Deutschland leben, das Kind selbst betreuen und erziehen und mit ihm in einem Haushalt leben. Während der Zeit, in der du Elterngeld erhältst, darfst du maximal 32 Stunden pro Woche arbeiten. Weitere Voraussetzungen sind, dass du im Jahr vor der Geburt ein steuerpflichtiges Einkommen von maximal 250.000 Euro hattest und gemeinsam mit dem anderen Elternteil nicht mehr als 300.000 Euro verdient hast. Den Antrag auf Elterngeld solltest du erst nach der Geburt deines Kindes stellen.

4. Question: Wie beantrage ich einen Personalausweis?

Context: Personalausweis erstmalig oder nach Ablauf beantragen  
Verfahrensablauf:\nSie müssen den Personalausweis persönlich bei der zuständigen Personalausweisbehörde beantragen. Als Deutsche mit Hauptwohnung im Ausland (Auslandsdeutsche) können Sie einen Personalausweis bei der Auslandsvertretung stellen in deren Bezirk Sie sich gewöhnlich aufhalten. Die zuständige Auslandsvertretung bestimmt das Auswärtige Amt. Ihren Antrag auf einen Personalausweis können Sie auch in Deutschland an jeder Personalausweisbehörde stellen. Diese Personalausweisbehörde wird als unzuständige Behörde Ihren Antrag bearbeiten wenn Sie einen wichtigen Grund dargelegen können. Ein solcher wichtiger Grund liegt z.B. dann vor wenn Sie geltend machen dass der Weg zur zuständigen Auslandsvertretung erheblich weiter ist als zur unzuständigen Personalausweisbehörde. Beachten Sie dass in diesen Fällen für die Bearbeitung Ihres Antrags eine Ermächtigung der zuständigen Auslandsvertretung

vorliegen muss und für die Antragstellung häufig eine Terminvereinbarung notwendig ist. Jugendliche ab 16 Jahren können den Personalausweis selbst beantragen. Kommen Jugendliche ab 16 und unter 18 Jahren ihrer Pflicht nicht nach muss der gesetzliche Vertreter beziehungsweise die gesetzliche Vertreterin den Antrag stellen. Gesetzliche Vertreter sind normalerweise die Eltern. Ein Elternteil kann sich bei der Antragstellung mit Vollmacht durch den anderen vertreten lassen. Ist der oder die Jugendliche 16 Jahre alt geworden muss der Antrag innerhalb von sechs Wochen gestellt werden. Für Kinder und Jugendliche unter 16 Jahren stellen beide Elternteile den Antrag gemeinsam wenn sie gemeinsam sorgeberechtigt sind. Die Kinder und Jugendlichen für die die Antragstellung erfolgt müssen immer persönlich erscheinen da die zuständige Stelle ihre Identität prüfen muss. Außerdem müssen sie unterschreiben wenn sie zum Antragszeitpunkt 10 Jahre oder älter sind. Seit dem 2. August 2021 ist die Aufnahme von Fingerabdrücken verpflichtend. Sie werden auf dem Ausweis-Chip gespeichert. Kindern unter sechs Jahren werden keine Fingerabdrücke abgenommen. Der Personalausweis wird zentral von der Bundesdruckerei in Berlin hergestellt. Sie erhalten als antragstellende Person vom Ausweishersteller einen PIN-Brief der eine fünfstelligen Transport-PIN für die Nutzung der eID-Funktion enthält. Außerdem werden Ihnen mitgeteilt: eine Entsperrnummer (PUK) zur Aufhebung der Blockierung nach dreimaliger Falscheingabe der PIN das Sperrkennwort für die eID-Funktion und das Vor-Ort-Auslesen sowie weitere Informationen zum Sperren der eID-Funktion. Bei Kindern und Jugendlichen unter 16 Jahren ist die eID-Funktion des Ausweises ausgeschaltet.

- T5<sub>BASE.OPEN</sub>: persönlich bei der zuständigen personalausweisbehörde
- T5<sub>BASE.CLOSED</sub>: mit der post (egal welcher herkunft) und mit der post (egal welcher herkunft)
- T5<sub>LARGE.OPEN</sub>: Sie müssen den Personalausweis persönlich bei der zuständigen Personalausweisbehörde beantragen.
- T5<sub>LARGE.CLOSED</sub>: nur bei Minderjährigen
- Luminous-base-control: Wird der Personalausweis erstmalig beantragt, muss der Antragsteller persönlich bei der zuständigen Personalausweisbehörde erscheinen. Als Deutsche mit Hauptwohnung im Ausland (Auslandsdeutsche) können Sie einen Personalausweis bei der Auslandsvertretung stellen, in deren Bezirk Sie sich gewöhnlich aufhalten. Die zuständige Auslandsvertretung bestimmt das Auswärtige Amt. Ihren Antrag auf einen Personalausweis können Sie auch in Deutschland an jeder Personalausweisbehörde stellen. Diese Personalausweisbehörde wird als unzuständige Behörde Ihren Antrag

bearbeiten, wenn Sie einen wichtigen Grund darlegen können. Ein solcher wichtiger Grund liegt z.B. vor, wenn Sie geltend machen können, dass der Weg zur zuständigen Auslandsvertretung erheblich weiter ist als zur unzuständigen Personalausweisbehörde. Beachten Sie, dass in diesen Fällen für die Bearbeitung Ihres Antrags eine Ermächtigung der zuständigen Auslandsvertretung vorliegen muss und für die Antragstellung häufig eine Terminvereinbarung notwendig ist. Jugendliche ab 16 Jahren können den Personalausweis selbst beantragen. Kommen Jugendliche ab 16 und unter 18 Jahren ihrer Pflicht nicht nach, muss der gesetzliche Vertreter beziehungsweise die gesetzliche Vertreterin den Antrag stellen. Gesetzliche Vertreter sind in der Regel die Eltern. Ein Elternteil kann sich bei der Antragstellung mit Vollmacht durch den anderen vertreten lassen. Ist der oder die Jugendliche 16 Jahre alt geworden, muss der Antrag innerhalb von sechs Wochen gestellt werden. Für Kinder und Jugendliche unter 16 Jahren stellen beide Elternteile den Antrag gemeinsam, wenn sie gemeinsam sorgeberechtigt sind. Die Kinder und Jugendlichen für die die Antragstellung erfolgt müssen immer persönlich erscheinen, da die zuständige Stelle ihre Identität prüfen muss. Außerdem müssen sie unterschreiben, wenn sie zum Antragszeitpunkt 10 Jahre oder älter sind. Seit dem 2. August 2021 ist die Aufnahme von Fingerabdrücken verpflichtend. Sie werden auf dem Ausweis-Chip gespeichert. Kindern unter sechs Jahren werden keine Fingerabdrücke abgenommen. Der Personalausweis wird zentral von der Bundesdruckerei in Berlin hergestellt. Sie erhalten als antragstellende Person vom Ausweishersteller einen "PIN-Brief", der eine fünfstellige "Transport-PIN" für die Nutzung der eID-Funktion enthält. Außerdem werden Ihnen mitgeteilt: eine Entsperrnummer (PUK) zur Aufhebung der Blockierung nach dreimaliger Falscheingabe der PIN das Sperrkennwort für die eID-Funktion und das Vor-Ort-Auslesen sowie weitere Informationen zum Sperren der eID-Funktion. Bei Kindern und Jugendlichen unter 16 Jahren ist die eID-Funktion des Ausweises ausgeschaltet.

- GPT-4: Um einen Personalausweis zu beantragen, musst du persönlich bei der zuständigen Personalausweisbehörde vorsprechen. Als Auslandsdeutscher kannst du deinen Antrag bei der für deinen Aufenthaltsort zuständigen Auslandsvertretung stellen. In Deutschland kann der Antrag bei jeder Personalausweisbehörde gestellt werden, wenn ein wichtiger Grund vorliegt. Jugendliche ab 16 Jahren können den Personalausweis selbst beantragen. Jugendliche unter 18 Jahren, die ihrer Pflicht nicht nachkommen, müssen den Antrag von ihrem gesetzlichen Vertreter stellen lassen. Wenn du unter

16 Jahre alt bist, stellen beide sorgeberechtigten Eltern den Antrag gemeinsam.

Bei der Antragstellung müssen Kinder und Jugendliche immer persönlich erscheinen, da ihre Identität geprüft werden muss. Seit August 2021 ist die Aufnahme von Fingerabdrücken verpflichtend, aber nicht für Kinder unter sechs Jahren.

Der Personalausweis wird von der Bundesdruckerei in Berlin hergestellt. Nach der Antragstellung erhältst du per Post einen "PIN-Brief" mit einer "Transport-PIN" für die Nutzung der eID-Funktion, eine Entsperrnummer (PUK), ein Sperrkennwort und weitere Informationen. Bei Kindern und Jugendlichen unter 16 Jahren ist die eID-Funktion ausgeschaltet.

5. Question: Wie beantrage ich Wohngeld?

Context: Wohngeld beantragen Verfahrensablauf:\nDas Wohngeld beantragen Sie am besten schriftlich. Nutzen Sie das oben stehende Papierformular. Sie erhalten es auch bei der zuständigen Stelle. Sie müssen unterschiedliche Papierformulare benutzen abhängig davon ob Sie Wohnraum gemietet haben (Mietzuschuss) oder dieser Eigentum von Ihnen ist (Lastenzuschuss). Wenn Sie einen Erstantrag auf Mietzuschuss stellen möchten können Sie auch den oben zur Verfügung stehenden Onlineantrag nutzen. Sie können Wohngeld auch formlos beantragen. Dabei gelten Besonderheiten. Ihr (formloser) Antrag gilt dann zu diesem Zeitpunkt als gestellt. Reichen Sie das ausgefüllte Formular samt aller erforderlichen Unterlagen innerhalb eines Monats nach erhalten Sie bei einer positiven Prüfung Ihres Antrags ab dem Tag der formlosen Antragstellung Wohngeld. Geben Sie Ihren Antrag und die erforderlichen Unterlagen erst später ab erhalten Sie Wohngeld erst ab diesem Datum. Die Entscheidung über Ihren Antrag erhalten Sie schriftlich in Form eines Bescheids. Bei einem positiven Bescheid überweist Ihnen die zuständige Stelle die Zahlungen einmal monatlich im Voraus auf das von Ihnen angegebene Konto.\n

- T5<sub>BASE\_OPEN</sub>: am besten schriftlich
- T5<sub>BASE\_CLOSED</sub>: mit deren hilfe ich einen lebensunterhalt leiste
- T5<sub>LARGE\_OPEN</sub>: Sie können einen Erstantrag auf Mietzuschuss stellen
- T5<sub>LARGE\_CLOSED</sub>: wenn Sie den Antrag auf Wohngeld stellen
- Luminous-base-control: Wird im Text erwähnt, wie man Wohngeld beantragt? Ja, es wird gesagt, dass man es am besten schriftlich beantragt. Wird auch erwähnt, ob man Wohngeld formlos beantragen kann? Ja, es wird erwähnt, dass dies möglich ist, aber es gibt Besonderheiten. Wird auch gesagt, wann

- man Wohngeld erhalten kann? Ja, es wird gesagt, dass man es ab dem Tag der formlosen Antragstellung erhält. Antwort: Wohngeld kann man am besten schriftlich beantragen. Man kann es auch formlos beantragen, aber es gibt Besonderheiten. Man erhält Wohngeld ab dem Tag der formlosen Antragstellung.
- GPT-4: Um Wohngeld zu beantragen, musst du das geeignete Papierformular ausfüllen, das du auch bei der zuständigen Stelle erhältst. Es gibt verschiedene Formulare, abhängig davon, ob du gemieteten Wohnraum hast (Mietzuschuss) oder dieser dein Eigentum ist (Lastenzuschuss). Für einen Erstantrag auf Mietzuschuss kannst du auch das zur Verfügung gestellte Onlineformular nutzen. Der Antrag kann auch formlos gestellt werden, wobei bestimmte Besonderheiten gelten. Dein Antrag gilt dann ab diesem Zeitpunkt. Das vollständig ausgefüllte Formular zusammen mit allen erforderlichen Unterlagen muss innerhalb eines Monats eingereicht werden, um bei positiver Prüfung ab dem Tag der formlosen Antragstellung Wohngeld zu erhalten. Wenn du deinen Antrag und die erforderlichen Unterlagen erst später einreichst, erhältst du das Wohngeld erst ab diesem Datum. Die Entscheidung über deinen Antrag erhältst du schriftlich in Form eines Bescheids. Bei einem positiven Bescheid überweist dir die zuständige Stelle die Zahlungen einmal monatlich im Voraus auf das von dir angegebene Konto.

6. Question: Was muss ich für die Geburt vorbereiten?

Context: Früherkennungsuntersuchungen für Kinder und Jugendliche wahrnehmen  
In Deutschland müssen Sie Ihr Kind elfmal ärztlich untersuchen lassen. U1: direkt nach der Entbindung U2: 3. bis 10. Lebenstag U3: 4. bis 5. Lebenswoche U4: 3. bis 4. Lebensmonat U5: 6. bis 7. Lebensmonat U6: mit 1 Jahr (10. bis 12. Lebensmonat) U7: mit 2 Jahren (21. bis 24. Lebensmonat) U7a: mit 3 Jahren (34. bis 36. Lebensmonat) U8: mit 4 Jahren (46. bis 48. Lebensmonat) U9: mit 5 Jahren (60. bis 64. Lebensmonat) J1: zwischen 13 und 14 Jahren Die Teilnahmepflicht an den Untersuchungen besteht unabhängig davon wie Eltern und Kind versichert sind. Sie ist im Kinderschutzgesetz Baden-Württemberg geregelt. Ziel der Untersuchungen ist die Früherkennung von Krankheiten Entwicklungsstörungen und Behinderungen die eine normale körperliche seelische oder geistige Entwicklung des Kindes gefährden. Therapien oder Förderungen können so rechtzeitig eingeleitet werden. Inhalt der Jugendgesundheitsuntersuchung (J1) sind zudem Themen wie Pubertät Sexualität Empfängnisverhütung schulische Entwicklung und gesundheitsgefährdendes Verhalten (Alkohol Rauchen Drogen). Die Untersuchung bietet Jugendlichen die Möglichkeit oft erstmalig ohne das Bei-

sein der Eltern Fragen zu gesundheitlichen und psychosozialen Themen beantwortet zu bekommen. Im Rahmen der Vorsorgeuntersuchungen finden auch eine Impfberatung und gegebenenfalls Impfungen statt. Der Bundesverband der Kinder- und Jugendärzte (BVKJ) empfiehlt als individuelle Gesundheitsleistungen weitere Früherkennungsuntersuchungen für Kinder und Jugendliche (U10 U11 J2). Sie erhalten dafür ein separates Checkheft.\n

- T5<sub>BASE\_OPEN</sub>: die entbindungsuntersuchungen für kinder und jugendliche
- T5<sub>BASE\_CLOSED</sub>: die vorgeburtliche („vorgeburtliche“) gabe des kindes, die „geburtsvorbereitung“ sowie die „vorbereitungsfeier“
- T5<sub>LARGE\_OPEN</sub>: die Früherkennung von Krankheiten Entwicklungsstörungen und Behinderungen
- T5<sub>LARGE\_CLOSED</sub>: die Früherkennungsuntersuchungen (U3, U9) und die damit verbundene Risikoauflärung
- Luminous-base-control: Wird im Text die Geburt erwähnt? Nein, es wird nicht erwähnt, was man für die Geburt vorbereiten muss. Antwort: Keine Antwort
- GPT-4: In Vorbereitung auf die Geburt sollten Sie darauf vorbereitet sein, dass Ihr Kind in Deutschland elfmal ärztlich untersucht werden muss, von der unmittelbaren Entbindung (U1) bis zu Untersuchungen in der Jugend, etwa zwischen dem 13. und 14. Lebensjahr (J1). Diese Untersuchungen sind unabhängig davon, wie Sie und Ihr Kind versichert sind und sind gesetzlich verpflichtend. Ziel ist die Früherkennung von Krankheiten, Entwicklungsstörungen und Behinderungen, die die normale Entwicklung des Kindes gefährden könnten. Auch Themen wie Pubertät, Sexualität, schulische Entwicklung und gesundheitsgefährdendes Verhalten werden behandelt. Eine Impfberatung und mögliche Impfungen sind ebenfalls Teil dieser Vorsorgeuntersuchungen. Der Bundesverband der Kinder- und Jugendärzte empfiehlt zusätzlich weitere Früherkennungsuntersuchungen (U10, U11, J2).

#### 7. Question: Was beinhaltet Infektionsschutzbelehrung?

Context: Infektionsschutzbelehrung inklusive Bescheinigung beantragen Ansteckende Krankheiten können über Lebensmittel auf andere Menschen übertragen werden. Wenn Sie bei Ihrer Arbeit mit Lebensmitteln in Kontakt kommen oder Gegenstände wie Geschirr reinigen soll das Risiko dieses Übertragungsweges minimiert werden. Daher brauchen Sie eine Bescheinigung des Gesundheitsamtes dass Sie an einer Belehrung über dieses Risiko teilgenommen haben.

Das Gleiche gilt wenn Sie sich regelmäßig in Küchen von Gaststätten oder sonstigen Gemeinschaftsverpflegungseinrichtungen aufhalten. Hinweis: Diese Bescheinigung ersetzt die früheren Gesundheitszeugnisse nach dem Bundesseuchengesetz. Wenn Sie bereits über ein Gesundheitszeugnis verfügen müssen Sie an keiner Belehrung teilnehmen. Die alten Zeugnisse behalten weiterhin ihre Gültigkeit. Vereinfachte Belehrung Für ehrenamtliche Helfer bei Vereinsfesten und ähnlichen Veranstaltungen reicht es wenn Sie das Merkblatt zur Vermeidung von Lebensmittelinfektionen lesen.\n

- $T5_{BASE\_OPEN}$ : eine bescheinigung des gesundheitsamtes
- $T5_{BASE\_CLOSED}$ : infektionskrankheiten wie hepatitis a und c
- $T5_{LARGE\_OPEN}$ : Bescheinigung
- $T5_{LARGE\_CLOSED}$ : Die Infektionsschutzbelehrung
- Luminous-base-control: Wird im Text die Belehrung über Infektionsschutz erwähnt? Ja, es wird erwähnt, dass man eine Belehrung über Infektionsschutz benötigt. Wird im Text erwähnt, was in der Belehrung enthalten ist? Ja, es wird erwähnt, dass man eine Bescheinigung des Gesundheitsamtes benötigt, dass man an der Belehrung teilgenommen hat. Antwort: Die Belehrung über Infektionsschutz beinhaltet eine Bescheinigung des Gesundheitsamtes, dass man an der Belehrung teilgenommen hat.
- GPT-4: Die Infektionsschutzbelehrung ist eine Schulung, die das Risiko der Übertragung von ansteckenden Krankheiten über Lebensmittel minimieren soll. Personen, die in ihrer Arbeit mit Lebensmitteln in Kontakt kommen oder Gegenstände wie Geschirr reinigen, benötigen eine Bescheinigung des Gesundheitsamtes, die belegt, dass sie an dieser Belehrung teilgenommen haben. Dasselbe gilt für Personen, die sich regelmäßig in Küchen von Gaststätten oder anderen Gemeinschaftsverpflegungseinrichtungen aufhalten. Für ehrenamtliche Helfer bei Vereinsfesten und ähnlichen Veranstaltungen ist eine vereinfachte Belehrung ausreichend, bei der sie ein Merkblatt zur Vermeidung von Lebensmittelinfektionen lesen.

#### 8. Question: Was muss ich bei einer Scheidung beachten?

Context: Scheidung beantragen Sie möchten sich scheiden lassen? Das Verfahren für Ehescheidungen findet vor dem Familiengericht statt. Als Antragstellerin oder Antragsteller müssen Sie sich durch eine Rechtsanwältin oder einen Rechtsanwalt vertreten lassen. Ihre Ehepartnerin oder Ihr Ehepartner benötigt keine rechtsanwaltliche Vertretung wenn sie oder er der Scheidung zustimmt und



selbst keine Anträge (zum Beispiel Anträge zum Unterhalt oder Zugewinnausgleich) stellen will. Scheidung beantragen Voraussetzungen:\nDie Ehe muss gescheitert sein. Eine Ehe ist gescheitert wenn die Lebensgemeinschaft der Eheleute nicht mehr besteht und eine Wiederaufnahme der Ehe nicht zu erwarten ist (Zerrüttung der Ehe). Dies wird vermutet wenn die Eheleute mindestens ein Jahr getrennt leben und beide die Scheidung wollen oder drei Jahre getrennt leben. Leben Sie noch nicht ein Jahr getrennt so kann das Gericht die Ehe nur scheiden wenn die Fortsetzung der Ehe für Sie aus Gründen die in der Person des anderen Ehegatten liegen eine unzumutbare Härte wäre.\n

- T5<sub>BASE\_OPEN</sub>: die Ehe muss gescheitert sein
- T5<sub>BASE\_CLOSED</sub>: Scheidungsfolgen
- T5<sub>LARGE\_OPEN</sub>: Die Ehe muss scheitern
- T5<sub>LARGE\_CLOSED</sub>: wenn die Ehe geschieden wird oder wenn der andere Ehegatte stirbt
- Luminous-base-control: Wird eine Scheidung eingereicht, muss man sich an das Familiengericht wenden. Welche Unterlagen man einreichen muss, hängt von den Umständen der Scheidung ab. Antwort: Die Belehrung über Infektionsschutz beinhaltet eine Bescheinigung des Gesundheitsamtes, dass man an der Belehrung teilgenommen hat.
- GPT-4: Wenn Sie eine Scheidung beantragen möchten, sollten Sie beachten, dass das Verfahren vor dem Familiengericht stattfindet. Als Antragsteller/in müssen Sie sich dabei durch einen Rechtsanwalt oder eine Rechtsanwältin vertreten lassen. Ihr Ehepartner benötigt keine anwaltliche Vertretung, sofern er der Scheidung zustimmt und keine eigenen Anträge (z.B. zum Unterhalt oder Zugewinnausgleich) stellt.

Eine wichtige Voraussetzung für eine Scheidung ist, dass die Ehe gescheitert ist. Dies wird angenommen, wenn die Lebensgemeinschaft der Eheleute nicht mehr besteht und keine Wiederaufnahme der Ehe zu erwarten ist (Zerrüttung der Ehe). In der Regel wird dies vermutet, wenn Sie und Ihr Partner seit mindestens einem Jahr getrennt leben und beide die Scheidung wollen, oder wenn Sie seit mindestens drei Jahren getrennt leben.

Wenn Sie noch nicht ein Jahr getrennt leben, kann das Gericht die Ehe nur scheiden, wenn die Fortsetzung der Ehe für Sie aufgrund von Gründen, die in der Person des anderen Ehepartners liegen, eine unzumutbare Härte darstellen würde.

9. Question: Was ist ein berufliches Gymnasium?



Context: Berufliches Gymnasium Berufliches Gymnasium:\nBerufliche Gymnasien sind Vollzeitschulen. Sie führen zur allgemeinen Hochschulreife. Darüber hinaus bieten sie eine gute Voraussetzung um anspruchsvolle Berufsausbildungen außerhalb der Hochschulen zu absolvieren. Berufliches Gymnasium der dreijährigen Aufbauform Das Berufliche Gymnasium der dreijährigen Aufbauform umfasst drei Schuljahre und endet mit der allgemeinen Hochschulreife. Das Angebot in Baden-Württemberg umfasst sechs verschiedene Richtungen die teilweise verschiedene Schwerpunkte beinhalten: agrarwissenschaftliche Richtung biotechnologische Richtung ernährungswissenschaftliche Richtung sozial- und gesundheitswissenschaftliche Richtung mit den Schwerpunkten Gesundheit und Soziales technische Richtung mit den Schwerpunkten Mechatronik Gestaltungs- und Medientechnik Informationstechnik Technik und Management Umwelttechnik wirtschaftswissenschaftliche Richtung mit den Schwerpunkten Wirtschaft Internationale Wirtschaft Finanzmanagement Berufliches Gymnasium der sechsjährigen Aufbauform Das Berufliche Gymnasium der sechsjährigen Aufbauform ist eine Vollzeitschule die in den Schwerpunkten Wirtschaft (6WG) Technik (6TG) sowie Ernährung Soziales und Gesundheit (6ESG) angeboten wird. Sie baut auf der Klasse 7 einer weiterführenden Schulart auf. Diese Form des Beruflichen Gymnasiums beginnt mit der Klasse 8. Mit dem Versetzungszeugnis in die Klasse 11 wird der mittlere Bildungsabschluss erworben.\n

- $T5_{BASE\_OPEN}$ : berufliche gymnasien sind vollzeitschulen. sie führen zur allgemeinen hochschulreife. darüber hinaus bieten sie eine gute voraussetzung um anspruchsvolle berufsausbildungen außerhalb der hochschulen zu absolvieren.
- $T5_{BASE\_CLOSED}$ : ein berufsbildendes gymnasium, in dem die schüler ihre kenntnisse und fertigkeiten unter beweis stellen
- $T5_{LARGE\_OPEN}$ : Vollzeitschulen
- $T5_{LARGE\_CLOSED}$ : das Berufliche Gymnasium mit den Fächern Wirtschaft, Technik und Naturwissenschaften
- Luminous-base-control: Wird im Text das Wort "Gymnasium" erwähnt? Ja, es wird im Text erwähnt, dass es sich um ein Gymnasium handelt. Wird im Text erwähnt, was ein berufliches Gymnasium ist? Ja, es wird erwähnt, dass es Vollzeitschulen sind, die zur allgemeinen Hochschulreife führen. Antwort: Ein berufliches Gymnasium ist eine Vollzeitschule, die zur allgemeinen Hochschulreife führt. Antwort: Die Belehrung über Infektionsschutz beinhaltet eine Bescheinigung des Gesundheitsamtes, dass man an der Belehrung teilgenommen hat.

- GPT-4: Ein berufliches Gymnasium ist eine Vollzeitschule, die zur allgemeinen Hochschulreife führt und gleichzeitig eine gute Voraussetzung für anspruchsvolle Berufsausbildungen außerhalb der Hochschulen schafft. Es gibt in Baden-Württemberg sechs verschiedene Richtungen, die jeweils unterschiedliche Schwerpunkte setzen, darunter agrarwissenschaftlich, biotechnologisch, ernährungswissenschaftlich, sozial- und gesundheitswissenschaftlich, technisch und wirtschaftswissenschaftlich.

Berufliche Gymnasien können in einer dreijährigen oder sechsjährigen Aufbauform besucht werden. Die dreijährige Form umfasst drei Schuljahre und endet mit der allgemeinen Hochschulreife, während die sechsjährige Form auf der Klasse 7 einer weiterführenden Schulart aufbaut und mit dem Übergang in die Klasse 11 den mittleren Bildungsabschluss verleiht.