

Neural Networks for Question Generation

Transformer based Question-Generation
of German Text

Research Paper

Bachelor's degree programme *Media Technology*
at the St. Pölten University of Applied Sciences

by:

Jaqueline Böck

mt181053

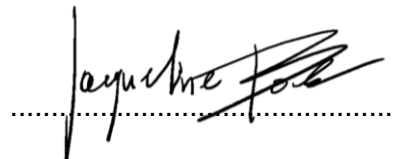
Supervising tutor: Dipl.-Ing. Armin Kirchknopf, BA MA BSc
St.Pölten, 08.07.2021

Declaration

- The attached research paper is my own original work undertaken in partial fulfilment of my degree.
- I have made no use of sources, materials or assistance other than those which have been openly and fully acknowledged in the text. If any part of another person's work has been cited, this either appears in inverted commas or (if beyond a few lines) is indented.
- Any direct citation or source of ideas has been identified in the text by author, date, and page number(s) immediately after such an item, and full details are provided in a bibliography (reference list) at the end of the text.
- I understand that any breach of the fair practice regulations may result in a mark of zero for this research paper and that it could also involve other repercussions.

St. Pölten, 08.07.2021

Place, Date

A handwritten signature in black ink, appearing to read 'Jaguhn', written over a horizontal dotted line.

Signature

Abstract

Due to the constantly growing amount of data in the field of linguistics, systems for automatic text analysis based on Neural Networks are becoming more and more popular. Since this is a technology that is capable of learning independently and becoming continuously more intelligent, it has been researched for years in Natural Language Processing (NLP) for a variety of applications. The sub-topics of text generation still represent a broad field of research. Especially automatic Question Generation (QG) from German text is still widely unexplored. Due to the enormous amount of information available in the digital age and the constantly growing or changing demands of society, the continuous education of each and everyone is an important prerequisite for the future. The proposed technology could be used to support the human iterative learning process in the form of automatically generating questions without the need of human input. In addition to personal interests of an individual, companies could also benefit from this technology. For example, the use of such systems for the creation of questionnaires for employee training would result in a saving of various resources such as time, effort and costs. In this paper, the current state-of-the-art of NLP will be reviewed, classical methods for text and Question Generation will be collected and their differences will be discussed. For generating questions from German text, special attention is paid to newer models called "Transformers". Based on the findings of previous research, an experimental implementation of such a system for the automatic generation of questions out of German texts using Transformers is developed and evaluated.

Kurzfassung

Auf Grund der stetig wachsenden Datenmengen im Bereich der Linguistik gewinnen Systeme zur automatischen Textanalyse, welche auf Neuronalen Netzen basieren immer mehr an Popularität. Da es sich hierbei um eine Technologie handelt, die fähig ist selbständig zu lernen und kontinuierlich intelligenter zu werden, werden diese in der Verarbeitung von natürlicher Sprache (Natural Language Processing/NLP) seit Jahren für eine Vielzahl von Anwendungsbereichen erforscht. Die Subthemen der Textgenerierung stellen hierbei nach wie vor ein breites Forschungsgebiet dar. Besonders die automatische Fragengenerierung aus Text als Inputmedium für deutschsprachige Textinhalte ist hier noch weitgehend unerforscht. Aufgrund des enormen Informationsangebotes des digitalen Zeitalters und der stetig wachsenden bzw. der sich ändernden Anforderungen an die Gesellschaft stellt die kontinuierliche Weiterbildung eines Jeden und einer Jeden eine wichtige Grundlage für die Zukunft dar. Diese Technologie könnte genutzt werden, um den menschlichen iterativen Lernprozess mit Hilfe von automatisch generierten Fragen zu unterstützen. Neben persönlichen Interessen einer Privatperson könnten auch Unternehmen von dieser Technologie profitieren. Der Einsatz solcher Systeme für die Erstellung von Fragebögen für Mitarbeiterschulungen würde etwa eine Ersparnis diverser Ressourcen wie Zeit, Aufwand und Kosten mit sich bringen. In dieser Arbeit wird der derzeitige Forschungsstand hinsichtlich des NLP beleuchtet, klassische Methoden zur Text und Fragengenerierung gesammelt und deren Unterschiede erarbeitet. Besonders den neuen Modellen, den „Transformern“ wird besonderes Augenmerk geschenkt. Der Fokus der Arbeit liegt im Generieren von Fragen aus deutschen Texten. Auf Basis des gefundenen Wissens wird eine experimentelle Implementierung und Evaluation eines derartigen Systems zur automatischen Fragengenerierung mittels Transformer durchgeführt.

Table of Contents

Declaration	II
Abstract	III
Kurzfassung	IV
Table of Contents	V
1 Introduction	1
2 Motivation	3
3 Method	4
4 Literature Review	5
4.1 Definition	5
4.2 Fields of Application	6
4.3 Language Models and Preprocessing	7
4.4 Classical/Traditional NLP Networks	8
4.4.1 Syntax-based Models	8
4.4.2 Semantic-based Models	8
4.4.3 Template-based and schema based Models	9
4.4.4 Rule-based Models	10
4.4.5 Statistical Models	10
4.4.6 Classic Neural Network Language Models	11
4.4.7 Recurrent Neural Network Language Models	12
4.5 Advanced Neural Networks for NLP (Transformer)	13
4.5.1 Definition	13
4.5.2 Overview of recent Applications using Transformer for QG	13
4.5.3 NLP for German Language	20
5 Automatic question Generation	21
5.1 Challenges and Difficulties	21
5.2 Question Types	21
5.3 Models for QG and recent Applications	22
5.3.1 Sequence-to-Sequence Models	22
5.3.2 Transformer based Models	22
6 Discussion of recent findings	24
7 Experimental Setup	25
7.1 Definition and requirements	25
7.2 Dataset and available Models	25
7.2.1 Used BERT Transformer	25
7.2.2 Used T5 Transformer	26

7.2.3	Used Hardware	27
7.3	Framework and Tools	27
7.4	Conceptual Design	28
7.5	Finetuning	29
7.5.1	T5 Finetuning Pipeline	29
7.5.2	Dataset preparation and preprocessing for finetuning	30
7.6	Question Generation	32
7.6.1	Question Generation Pipeline	32
7.6.2	Dataset preparation and preprocessing for QG	33
8	Evaluation and Results	36
8.1	Technique and Metrics	36
8.2	Input Texts	36
8.3	Evaluation Setup	38
8.4	Participants	39
8.5	Results	41
9	Discussion of the Experiment	43
10	Conclusion and Future Work	44
	References	46
	List of figures	51
	List of tables	52

1 Introduction

Many of the already existing methods for generating questions are not only using standalone texts as input. Often other additional inputs are required (e.g., predefined keywords). This additional knowledge can be provided by humans. Other well-known systems are designed to generate questions that match predefined answers. In both scenarios, a model tries to find the specific answer or the predefined keyword in the input sentences/text. Afterwards, sentences including the keyword are paraphrased to more question-like ones.

However, generating questions from pure text without additional human input is a much more difficult task since a broader knowledge of the context is required. Another challenge is that although datasets and neural architectures for NLP tasks are freely available, architectures for German settings are relatively rare. Moreover, these German models have only been developed for very limited scenarios. Especially the number of approaches for German Question Generation is little.

The main goal of this thesis is to generate questions from German texts without the need of additional human inputs like keywords. The aim is to find a way to generate those questions by using Neural Networks. This thesis should provide an implementation based on existing models for the use of automatic, domain-independent generation of questions.

The latest generation of natural language algorithms for NLP tasks are the so-called "Transformers". Transformers such as the BERT model (Bidirectional Encoder Representation from Transformers) or GPT models (Generative Pretrained Transformer) have become popular in recent years due to their great flexibility. These architectures seem to be good alternatives compared to previously used classical methods since those can deal with a wider range of NLP tasks. Those newer models are carried out and discussed within the first half of this thesis.

For German texts, German or at least multilingual models are required. Concrete examples of these pretrained models are Deepset BERT¹ and Googles multilingual

¹ <https://www.deepset.ai/german-bert>, retrieved on 08.07.2021.

BERT². In addition, Transformers offer the possibility of not only using pretrained models, but also to adapt these basic models to the requirements which are needed for more specific tasks. This process is known as "fine-tuning". This ability makes the use of Transformers particularly interesting for the implementation of a German all-purpose QG system.

The second half of this thesis includes an experimental implementation of one possible way of generating questions from German text.

Moreover, taking all the previously mentioned topics in consideration, following research questions arise:

- How do Transformer architectures differ from conventional machine learning systems for NLP and what are their advantages and disadvantages?
- Which datasets, (pretrained) models and combinations of those can be used to generate questions from German text?
- How can complex NLP systems be broken down into smaller components to create more simple architectures for Question Generation?
- How can an architecture for Question Generation of German text be practically implemented?

At the beginning of the paper, an overview of different tasks where Natural Language Processing can be applied and has been used in recent years is given, followed by the most relevant preprocessing steps. Relevant classical NLP models are briefly explained and newer neural models are introduced.

Subsequently, possible occurring challenges and difficulties of these systems are outlined. Furthermore, an overview of different possible question types is given. After examining various previous implementations of such systems, a practical implementation of such a system for German texts was developed and evaluated. The proposed system got evaluated by human judgment and occurring issues are stated. An insight on the implementation of the evaluation procedure is given. The results are discussed, the findings are summarized and ideas for further approaches are presented at the end of the paper.

² <https://huggingface.co/bert-base-multilingual-cased>, retrieved on 08.07.2021

2 Motivation

Digital solutions for problems and/or processes are becoming increasingly relevant in today's world. One big factor is the impact of the digital revolution. In business as well as in science, it is important to always stay updated in terms of the latest developments to stay relevant.

The use of artificial intelligence can bring enormous benefits in the realization of such tasks. For example, such systems can systematically support humans and/or even be responsible for certain tasks. This can help to save resources, which can efficiently be used for other, more complex areas where digital solutions are not applicable. Especially the use of artificial intelligence regarding language and text offer great potential to enhance such processes actively. Natural Language Processing has the potential to make the fast pace of these developments more manageable. Especially the automatic generation of questionnaires has become more interesting in recent years. Cost savings in terms of staff time are particularly interesting in this context, since the resources saved can be used for more complex tasks.

English solutions for such systems are actively researched since several years. Nevertheless, particularly for the German language, such applications can rarely be found. However, considering that a significant part of the areas that are important for digitization are speaking German, whether it is in large companies for employee training or in smaller settings such as education, it is important to invent such a German solution. With regards to the points mentioned above, the goal of this paper is to give a basic idea of how such a German solution could be designed and implemented.

3 Method

Initially, an extensive research based on previous applications for Question Generation from texts using machine learning techniques is carried out. Google and Google Scholar were mainly used for this task. In addition, various online archives, libraries, and journals such as arXiv.org, link.springer.com, ijcseonline.org and towardsdatascience.com were used during the research. Huggingface.co and Github.com were essential resources for the code, datasets and models.

The main keywords/phrases used for the search include: Natural Language Processing, Question Generation, Transformers, Finetuning, Text Generation, Question Generation Pipeline and Framework

Subsequently, information and ongoing resources on current state-of-the-art Transformers, (german/multilingual) pretrained models and datasets that can be used for finetuning/training those for German Question Generation will be introduced. The advantages and disadvantages of recent as well as current state-of-the-art methods are presented and possible combinations of those for generating a proper pipeline to achieve the stated goal are discussed. Finally, regarding these findings, one possible framework is practically implemented and afterwards evaluated by human assessment. For the evaluation, questions were generated based on three texts with the difficulty levels: easy, moderate and difficult. For the human assessment a questionnaire was prepared in which the generated questions were evaluated accordingly to predefined criteria. This questionnaire was distributed on social media such as Facebook and Twitter as well as in several WhatsApp groups.

4 Literature Review

In this section, the basic principles of Natural Language Processing (NLP) are presented. The use of NLP in various fields of application is described and an overview of the most important implementation steps is given. At the end of this section, the already well-established classical methods that can be used for Question Generation are discussed followed by the most recent ones including state-of-the-art models called “Transformers”.

4.1 Definition

Natural Language Processing (NLP) is a technology combining the disciplines of Artificial Intelligence, Linguistics, and Computer Science. These subtopics include Deep Learning and Neural Networks. Using algorithms, computers can work with huge amounts and different types of data. In terms of NLP, the semantic of the human language in form of text or speech is learned by the software and enables the machine to automatically manipulate the input data in a way that meaningful outputs depending on the given tasks can be generated. Since natural language is astoundingly complex and diverse, the main purpose for NLP is it to process and filter these given chunks of data in a to make it possible to work with the meaningful content. This process provides the possibility to make the computer-human interaction as easy and efficient and possible (Jain et al., 2018). For achieving that, NLP systems break down the complex input data into smaller, more expressive segments to retrieve and understand the important information better.

4.2 Fields of Application

There is a huge variety of techniques that can be used for understanding and interpreting human language. Hence the variety of human language is extremely great, many different techniques are needed to cope with the multitude of specific tasks. The broad field of NLP extends from statistical to rules-based and machine learning approaches, up to algorithmic techniques.

Important areas of application in which NLP is used are:

- **Content categorization:**
It is the process of automatically analyse linguistic input and structuring it into organized groups. Content summarization, indexing and content alerts are used for achieving that.
- **Topic discovery and modelling:**
It is used for retrieving hidden semantic structures in a textual corpus as well as for applications where optimization and forecasting are needed. It is based on a statistical approach.
- **Contextual extraction:**
Is used for breaking down complex sentences into smaller elements (called “n-grams”) for automatically retrieving more structured themes and facets out of unstructured textual data.
- **Sentiment analysis:**
Is the process of determining the emotional context like the mood or subjective opinions stated in huge documents.
- **Speech-to-text and text-to-speech conversion:**
Transforming spoken words into written text, and vice versa.
- **Document summarization:**
Automatic recognition of the important information and display of the content in a shortened form.
- **Machine translation:**
Automatic conversion of text or speech from the input to the output language.³

³ https://www.sas.com/en_us/insights/analytics/what-is-natural-language-processing-nlp.html ,
retrieved on 29.12.2020.

4.3 Language Models and Preprocessing

Language Models (LMs) are systems which interpret the input data using algorithms (Oberoi, 2020). LMs are important especially in the beginning of NLP since they form the basis for a variety of NLP applications.

Common use cases for Language Models are applications for handwriting recognition, Machine Translation tasks by calculating the probabilities of future outputs, Speech Recognition, where it is accomplished to predict the next word in a sentence (Jing & Xu, 2019), Part-of-Speech-Tagging (POS), Parsing and Information Retrieval, up to Optical Character Recognition and many other utilizations (Anand, 2020). Working with Natural Language Data (especially handwriting) is extremely challenging and time-consuming due to its variety of linguistic phenomena (Jing & Xu, 2019). More information and explanations of the most common LMs are provided at section *4.4 Classical/Traditional NLP Networks* and *4.5 Advanced Neural Networks for NLP (Transformer)*. Before specific LMs can be applied on linguistic corpora the existing data often needs to be preprocessed depending on the specific tasks.

Text preprocessing can be defined into 3 main steps:

- 1. Tokenization/Text Segmentation:**

Is the processes of splitting long sequences of text into shorter segments.

- 2. Normalization:**

The goal is to equalize the given input data so that the data can be processed uniformly.

Common steps are:

- **Stemming:** obtaining the word stem by removing affixes
- **Lemmatization:** based on a word's lemma. Goal is to capture the canonical forms of a word
- Setting all characters to lowercase
- Stripping white space
- Removing numbers, punctuation, stopwords and sparse terms

- 3. Noise Removal:**

Accomplished by removing:

- Text file headers, footers, regular expressions
- HTML, XML, etc. markups
- Metadata
- Extracting data from databases or other formats like JSON

(Mayo, 2017)

4.4 Classical/Traditional NLP Networks

In the course of time, there have been overlaps in the development of the various methodologies for generating questions from text. This makes it difficult to differentiate clearly between them. Apart from that, hybrid systems have been and still are developed. Regardless, this chapter attempts to classify the methods into syntax-based, semantic-based, template-based, schema-based, rule-based, statistical, and classical Neural Language Models. Neural LMs outperformed the rule-based and statistical ones in recent applications due to their abilities and effectiveness (Anand, 2020).

Accordingly to the papers of Yao et al., 2012, Lindberg, 2013 and Kurdi et al., 2020, the different methods used for Question Generation can be categorised as followed:

1. **Syntax-based**
2. **Semantic based**
3. **Template based**
4. **Rule-based**
5. **Schema-based**
6. **Statistical Methods**
7. **Learning/Neural-based**

4.4.1 Syntax-based Models

Syntax-based LMs are often consisting of a variety of processing steps to understand the syntax of the input (e.g., text). Common steps that are used are Sentence Simplification, Keyphrase Identification and finding Distractors (Lindberg, 2013) depending on their Parts of Speech (POS) (Kurdi et al., 2020). Also baseline methods like Latent Semantic Analysis (LSA) in combination with other parameters like knowledge-based methods can be applied (Hutchison et al., 2010).

4.4.2 Semantic-based Models

Semantic models dig deep into the semantic level of an input. For achieving good results, additional knowledge like ontologies and taxonomies are often required. An example of this model can be found in the paper of Lindberg, 2013, where Semantic Role Labelling (SRL) is used for solving this issue.

4.4.3 Template-based and schema based Models

These LMs are characterised by offering pre-defined text. These texts also include some placeholders that have been generated from the input. The placeholders can be replaced by syntactic and/or semantic features of entities (Kurdi et al., 2020). On one hand, this approach makes it possible to ask questions with arbitrary wording that are little depending on the source text. On the other hand, the types of questions that can be produced are limited (e.g., what, who, when, where). Grouping templates that represent alternative approaches for the same issue are considered as schema-based models (Lindberg, 2013). An example of such a template and its results are shown in *Figure 1: Example of category-pattern-template block* (Lindberg, 2013). and *Figure 2: Examples of template-based questions* (Lindberg, 2013).

```
<category>
  <pattern>
    <S>
      <NP person="true">_person_</NP>
      <VP> <VBD>went</VBD>
        <PP> <TO>to</TO> <NP location="true">_place_</NP> </PP>
      </VP>
    <star />
  </S>
</pattern>
<template> Where did _person_ go? </template>
<template> Who went to _place_? </template>
<template> Why did _person_ go to _place_? </template>
</category>
```

Figure 1: Example of category-pattern-template block (Lindberg, 2013).

The boy went to school.

Where did The boy go?

Who went to school?

Why did The boy go to school?

Figure 2: Examples of template-based questions (Lindberg, 2013).

4.4.4 Rule-based Models

Rule-based models use knowledge to interpret and manipulate given data using predefined rules available as regular expressions or dictionary-based keywords. These sets of rules are often human crafted. In terms of decision making, the output of these approaches is completely dependent on the given input rules. In rule-based models, one common process for extracting the interesting data out of the text corpora is Named Entity Recognition (NER) (Nguyen et al., 2016).

4.4.5 Statistical Models

Statistical systems can predict the next word of a sequence by developing mathematical probabilistic models on the given input text. Traditional linear models are relatively low in their feature dimensions since every single feature gets represented as one defined word embedding. To improve the dimensionality of such systems it is a common approach to implement combinations of different features manually to the frameworks (Goldberg, 2015).

The most common models used are the n-gram models which are explained in the next paragraph. These also tend to be the simplest and most fundamental ones in the history of NLP (Le, 2019). Other common techniques are Part of Speech Tagging (POS) (Tiet et al., 2018), Topic Modelling (especially using Latent Dirichlet Allocation (LDA) (Kapadia, 2019) and Term Frequency-Inverse Document Frequency (TF-IDF) (Kumawat, 2020).

4.4.5.1 N-Grams

The n-gram model is the most primal model in the history of NLP. N-grams are sequences of n words. The functionality of this statistical approach is based on the amount and the frequency of the different n-grams that are used in the input text. Giving the example sentence “She is driving her...”, the attended n-grams would be:

- unigrams: “She”, “is”, “driving”, “her”
- bigrams: “She is”, “is driving”, “driving her”
- trigrams: “She is driving”, “is driving her”
- 4-grams: “She is driving her”

(Le, 2019)

The quality of a probabilistic model can be measured by its perplexity. The perplexity is always depending on its input corpus. Comparing two probabilistic models, lower scores in perplexity indicate a better performance. Nevertheless, the drawback of n-gram-LMs is significant since these models tend to calculate probabilities that never appeared in

the training data. These circumstances do not represent the actual situation. For solving this problem, smoothing techniques can be applied (Jing & Xu, 2019).

Moreover, the most interrupting problem using n-grams is the sparsity problem. For larger documents which come with an huge amount of words, the accuracy of these simple models get harmed (Le, 2019).

4.4.6 Classic Neural Network Language Models

Neural network models like Feed Forward Neural Networks (FFNNs) use word embeddings for decision making. More advanced Networks are the Recurrent Neural Networks (RNNs). These provide the possibility to capture the discourse of input sentences, corpuses and subwords (Le, 2019).

Compared to sparse-input linear models which present the feature embedded into defined dimensional space, neural models represent the features as dense vectors. Since two vectors can be very similar, for example, comparing the learned vectors from the word “cat” with those of the word “dog”, these models are able to filter important features by their own using statistical assumptions. Just as words, also distance and positional features are represented as vectors. This makes it possible to differ between similar words with different meanings depending on the position of the word in a sentence. Another benefit of these neural models is the fact that these networks only need to focus on the core features since they can find the indicative features and combinations of those by their own. Although the high dimensionality offers many advantages, there is a risk that these systems will consume unnecessarily long computation time and data resources. Therefore, it is important to choose the number of features carefully (Goldberg, 2015).

4.4.6.1 Feed Forward Neural Probabilistic Language Models

Since Feed Forward Neural Networks (FFNNs) (Lavine & Blank, 2009) use fixed-length inputs, they are not well suited for working with variable-length data. The given information gets processed straight forward from one input to another hidden- or output layer in one direction. For predicting the next word, the previous context as n-1 words is considered. For well-functioning fully connected Neural Networks, it is important that many trainable parameters are used. Nevertheless, the number of training data is limited for these models, this poses an additional problem. Moreover, these architectures suffer from severe memory loss because they rely on time. Because of that, they have some significant limitations in predicting what sequence is coming next since they cannot remember the past except of the data they have been trained by. These circumstances make FFNNs inefficient and expensive (Jing & Xu, 2019).

4.4.7 Recurrent Neural Network Language Models

4.4.7.1 Basic RNN Models

Recurrent Neural Networks (RNNs) (Sherstinsky, 2020) have an Encoder and a Decoder. This seq2seq models can also be build more advanced by adding specific attention mechanisms which connect the included Encoder and the Decoder with each other (Vaswani et al., 2017). Even though these architectures can be built robust, they still face the data-length and time-problem of FFNNs. By parameter sharing, duplicated calculations get avoided as the input window shifts. Even though these architectures also consider time information, which make them better suited for long-term dependencies, it is still challenges to generate a good outcome. Another challenging aspect is that during training, the parameters gradients might explode or disappear which causes infinite parameter values and slow training speed (Jing & Xu, 2019).

4.4.7.2 LTSM-RNN Models

By adding Long Short-Term Memory (LSTM) (Hochreiter & Schmidhuber, 1997) to RNN Models, the memory-loss problem can be controlled. This is achieved by its three-gate structure (input, output and forget gate). The distributed word in this model gets normalised by a Softmax-layer. The Softmax-layer makes the algorithm considering all given words of the input data. Because of the richness of the data and when focusing on right vocabulary, these approaches are very time-consuming (Jing & Xu, 2019).

4.5 Advanced Neural Networks for NLP (Transformer)

4.5.1 Definition

The Transformer in NLP is a novel architecture that use attention and recurrence to handle dependencies between the input and the output sequences. More precisely, Transformer use parallelized attentive mechanisms for encoding and decoding which allows them to deal with long-range dependencies. By that, Sequence-to-Sequence (seq2seq) tasks especially for long input texts can be done with ease (Prateek, 2019).

4.5.2 Overview of recent Applications using Transformer for QG

In recent years, many robust Question Generation (QG) architectures have been developed. Traditional Sequence-to-Sequence approaches using RNNs like LSTM or GRU (Gated Recurrent Unit) have been very popular. Examples are proposed in the papers of Dong et al., 2019 and Zhou et al., 2018. Also, Transformers have been used for encoding the source context for longer texts like documents or paragraphs. The use of additional linguistic features have produced good outcomes in recent applications. Especially the use of answer-aware inputs which are related to the context and/or approaches which combine these proposed techniques have achieved state-of-the-art results in recent years (Dong et al., 2019; Du et al., 2017; Zhou et al., 2018). Lopez et al., 2021 proposed a Transformer based QG system. The architecture doesn't rely on any additional features by using only one pretrained Transformer language model (GPT-2). Chan & Fan, 2019 have introduced three neural architectures based on top of a pretrained BERT language model for Question Generation task. Since the BERT model is build out of multiple stacked Transformer-blocks using self-attention, a recurrence architecture which draws global dependencies between input sentences can be achieved.

This ability is from high interest since well-formed questions rely on a wider knowledge of the input. Especially when it comes to dealing with longer paragraphs. Ramsri Goutham Golla published an application for Multiple Choice Question Generation using a BERT Transformer for detecting and summarizing the most important sentences out of a longer textual input⁴. After Tokenization of the sentences and Keyword Extraction

⁴ https://github.com/ramsrigouthamg/Generate_MCQ_BERT_Wordnet_Conceptnet, retrieved on 17.02.2021.

using the nltk^{5,6} and the flashtext⁷ framework, fill-in-the-gap sentences get created by removing the specific keywords. Distractors are generated using Conceptnet⁸ and Wordnet⁹. This shared application was a fundamental resource for this thesis. Moreover, Golla also developed an open-source application called „Questgen AI“¹⁰ which focuses on easy to use QG-Algorithms for English text. Another Question Generation framework was introduced by Suraj Patil¹¹. In this resource a T5 Transformer model trained on the SQUADv1¹² dataset which has been introduced by Rajpurkar et al. in 2016, has been trained for solving different Question Generation tasks like answer-aware Question Generation, Answer Extraction as well as end-to-end QG and Multitask Question Answering and Generation (QA/QG).

There is already a wide range of available Transformers for many different NLP tasks available. BERT, T5 and GPT were used for a variety of NLP tasks in recent years especially with regards to QA/QG.

The website Huggingface.co¹³ categorizes these models as followed:

- Autoregressive models
- Autoencoding models
- Sequence-to-Sequence models
- Multimodal models

For more information of the different models please refer to the provided link. The used models in this paper are a T5- and a BERT-Transformer. The functionality of those is described below.

⁵ <https://www.nltk.org/api/nltk.corpus.html>, retrieved on 16.05.2021.

⁶ <https://www.nltk.org/api/nltk.tokenize.html#module-nltk.tokenize>, retrieved on 16.05.2021.

⁷ https://flashtext.readthedocs.io/en/latest/keyword_processor.html, retrieved on 16.05.2021.

⁸ <https://conceptnet.io/>, retrieved on 24.01.2021.

⁹ <https://www.nltk.org/search.html?q=wordnet>, retrieved on 24.05.2021.

¹⁰ <https://github.com/ramsrigouthamg/Questgen.ai>, retrieved on 15.04.2021.

¹¹ https://github.com/patil-suraj/question_generation, retrieved on 07.04.2021.

¹² <https://deepai.org/dataset/squad1-1-dev>, retrieved on 17.05.2021.

¹³ <https://huggingface.co/transformers/summary.html>, retrieved on 17.05.2021.

4.5.2.1 BERT

The BERT-Transformer (Bidirectional Encoder Representation from Transformers) which architecture is illustrated in *Figure 3: Transformer Architecture* (made by author, based on Neptune.ai). is a model which consists of an Encoder and a Decoder.

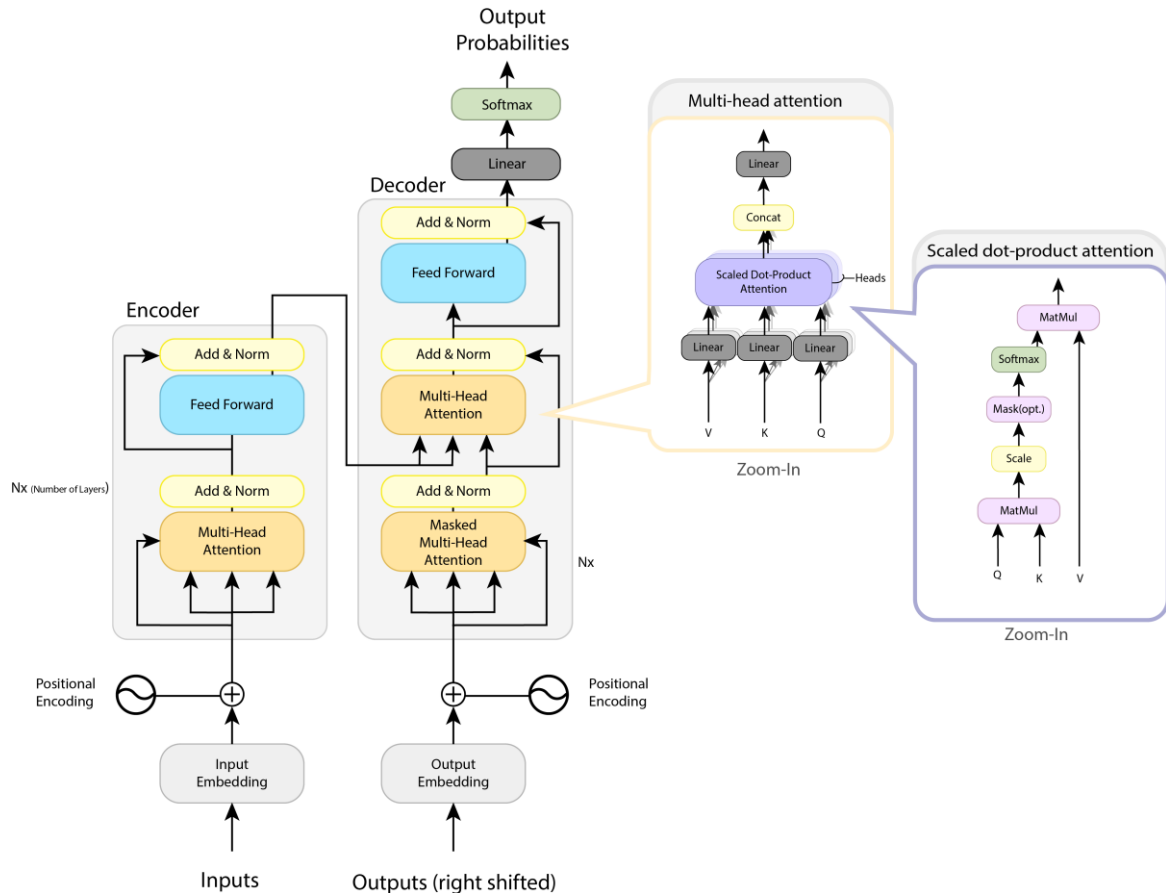


Figure 3: Transformer Architecture (made by author, based on Neptune.ai¹⁴).

The Encoder can be built out of a multiple stack of Transformer-Encoder layers which include fully-connected Neural Networks called “heads” (Horev, 2018). Each of these heads consists of a Neural Network augmented self-attention mechanism (Rogers et al., 2020). If a sequence of tokens is put into a BERT model, the tokens get embedded into a vector space. This modality is essential for setting different weights on the given input tokens (Horev, 2018). Furthermore, BERT consists of a fully connected layer at the end of the architecture.

¹⁴ <https://neptune.ai/blog/bert-and-the-transformer-architecture-reshaping-the-ai-landscape>, retrieved on 07.07.2021.

An illustrative overview of the connected layers is provided in *Figure 4: BERT Fully-Connected Layer* (made by author inspired by ai.googleblog.com).

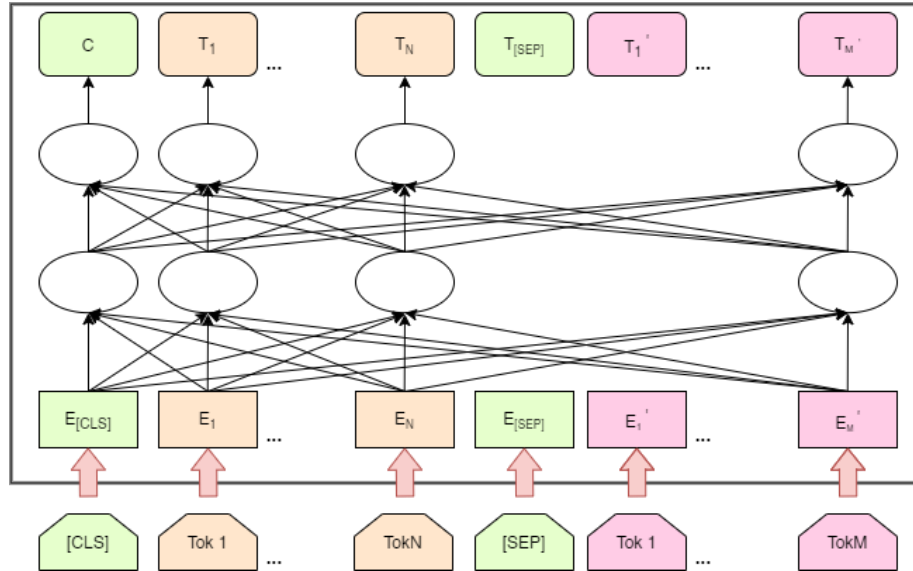


Figure 4: BERT Fully-Connected Layer
(made by author inspired by ai.googleblog.com¹⁵).

This fully connected layer combines each of those different heads. A skip connection is added to each of this layers and also a normalization procedure is applied at the end (Rogers et al., 2020). Because of its attention mechanism, BERT can learn contextual relationships between words in a sentence or text (Horev, 2018). Hence BERT can read the entire given input sentence at once, the model can learn the context of an input by considering its surroundings. Since knowledge from the left as well as from the right side of a word are targeted, the architecture of BERT is considered as bidirectional (Horev, 2018).

Moreover, as BERT can predict tokens in a bidirectional way, the pretraining of the BERT Transformer is based on two semi-supervised tasks:

1. Masked Language Modelling (MLM): It is the process of predicting randomly masked tokens out of the given input. For that, a certain number of words of the input sentence get replaced with a [MASK] token. The goal of MLM is to predict the word that has been replaced.

¹⁵ <https://ai.googleblog.com/2018/11/open-sourcing-bert-state-of-art-pre.html>, retrieved on 07.07.2021.

2. Next Sentence Prediction (NSP): Two sentence pairs are fed into the Transformer and BERT must predict if the second sentence is the subsequent sentence in of the first input sequence. The distribution of the input data normally consists of 50% correlating sentence pairs and 50% random sentences from the given corpus. The random sentence should be disconnected to from the first word span to help the model distinguishing between two sentences during the training procedure. For that, the use of special tokens helps the architecture to process the input tokens. At the beginning of a sample a [CLS] token is set whereas at the end an [SEP] token is added. As already mentioned and illustrated in *Figure 5: BERT Tokenization* (made by author, based on towardsdatascience.com)., adding embeddings to these tokens helps the algorithm to distinguish between two sentences (for example “Sample A” and “Sample B”). For monitoring the position in the sequence, also a positional embedding for each token is added (Horev, 2018).

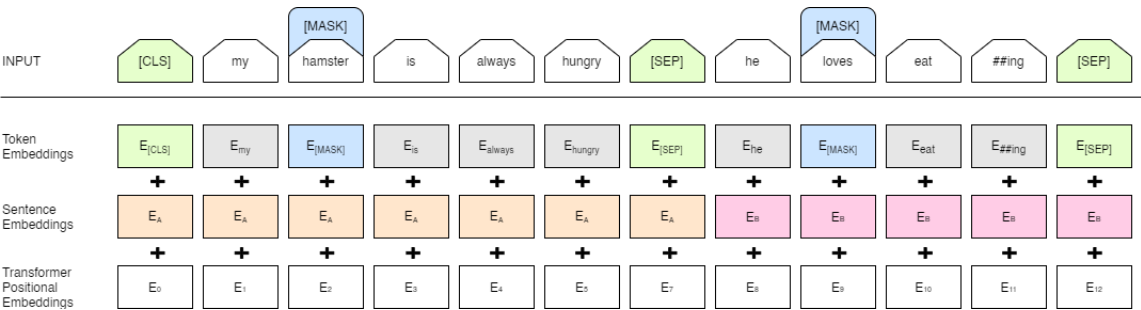


Figure 5: BERT Tokenization
(made by author, based on towardsdatascience.com¹⁶).

For getting to know if the second sentence is really related to the first one, the sequences must run through the whole Transformer-model first. Afterwards a simple classification layer must be passed and the probability for the next sequence gets calculated by using Softmax. In training, MLM and NSP are executed simultaneously for minimizing the loss function (Horev, 2018).

¹⁶ <https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270>, retrieved on 07.07.2021.

BERT can be finetuned for downstream applications by adding one or more fully-connected layers on top of the final Encoder layer (Rogers et al., 2020).

Examples of the use of BERT are:

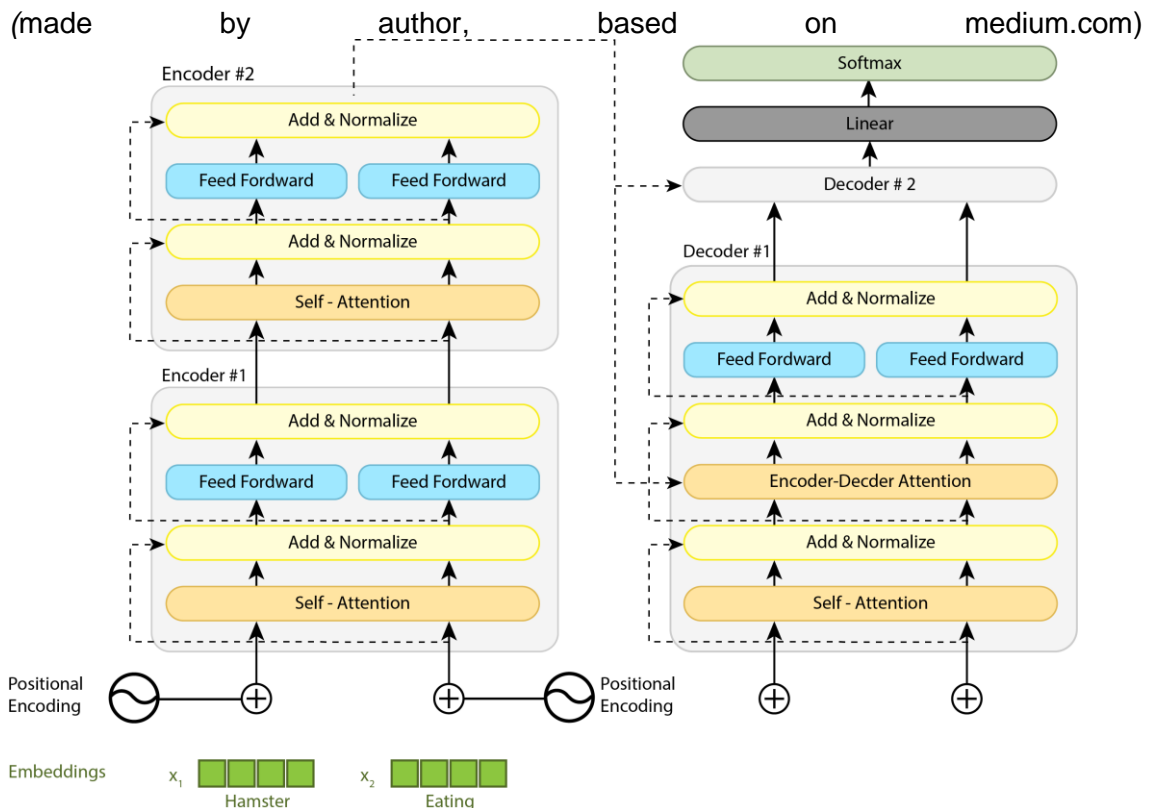
- Classification
- Question Answering (QA)
- Summarization
- Named Entity Recognition (NER)

(Horev, 2018)

4.5.2.2 T5 Transformers

The T5 model combines multiple NLP tasks to one unified text-to-text format (Raffel et al., 2020). It is an Encoder-Decoder based model which is pretrained on unsupervised and supervised tasks.

A detailed illustration of the architecture is shown in *Figure 6: Architecture T5 Transformer*



*Figure 6: Architecture T5 Transformer
(made by author, based on medium.com¹⁷).*

T5 is a seq2seq model that can be used for any common NLP applications. The model can be easily modified and comes with a great flexibility when it comes to applying it to different tasks (Raffel et al., 2020).

¹⁷ <https://medium.com/analytics-vidhya/t5-a-detailed-explanation-a0ac9bc53e51>, retrieved on 07.0.2021.

Examples usages of T5 are:

- Document Summarization
- Machine Translation
- Question Answering and Classification
- Regression Tasks

The training objective of T5 is very similar with those of BERT by using bidirectional Masked Language Modelling. This makes it possible to derive the representation of the masked tokens anytime in either the left or the right context of the word in a sentence. The difference to the proposed BERT method is that in T5, multiple consecutive tokens are replaced with one single [MASK] keyword instead of just one word per [MASK] keyword as it is in BERT. Therefore, the inputs and the outputs of the proposed T5 model are always strings and not just single words.

Finetuning can be easily applied by showing the model the input and the output text pairs that should be used. Each input is highlighted by adding task-specific prefix-text¹⁸.

4.5.3 NLP for German Language

Even though in recent applications the usage of Transformers for German language mostly focuses on German translation tasks, there is a great opportunity for implementing a German architecture for other tasks like QA and QG. Since there are already certain pretrained models for German as well as for multilingual tasks available, finetuning those could have great potential for solving more difficult German tasks. Especially the proposed BERT model and the T5 model seems to provide great advantages for such implementations.

¹⁸ <https://prakhartechviz.blogspot.com/2020/04/t5-text-to-text-transfer-transformer.html>,
retrieved on 14.04.2021.

5 Automatic question Generation

5.1 Challenges and Difficulties

In contrast to common NLG applications where often only the semantics of the text must be understood (e.g., translation), in Question Generation it is also important to understand the meaning of the text to be able to generate meaningful questions. Accordingly, simple one-to-one correlations can be applied here, as is the case of simple translation tasks (Kurdi et al., 2020). Moreover, generating question is understood as an “abstractive” task because the outcome can be, but is not necessarily present in the input. In addition, another challenge that comes with QG is detecting silly question that are framed of the variety of dependencies and relations along phrases and words. It is essential to differ between those nonsensical and meaningful ones (Padmanabhan, 2020).

The majority of recent approaches for generating questions are architectures where the keyword or answer is provided in the first place. In these approaches, the scope of the questions and the types of the questions are limited. Because of that, a QG system requires a certain understanding of the input content and the semantic relationships among sentences to rephrase words and rearrange text in a senseful way. Therefore, paying attention in correct syntax and grammar is important (Lindberg, 2013). For that, the understanding of long and multiple sentences is essential. For solving those issues, the use of Transformers can be revolutionary.

5.2 Question Types

As already mentioned earlier, the generation of questions using machine learning comes with great challenges. Additionally, the variety of question types is also a major challenge in the development of building robust systems. The most important types of questions are:

- **Gap-fill questions:** can be constructed with relatively little effort.
- **Wh-questions** (where, what, when, who, whose, whom, which, why and how)
- **Multiple choice questions (MCQ):**

Issues in generating MCQ are:

- A lack of distractors in textual data
- Additional/external knowledge and/or structural knowledge sources are needed to differ between true and false.

- **Jeopardy-style questions** (e.g., Quizzes)
- **Mathematical word problems**
- **Questions from case-based knowledge:**
Problem: uncommon words (unknown stem content and verbalisation)

(Kurdi et al., 2020)

5.3 Models for QG and recent Applications

5.3.1 Sequence-to-Sequence Models

Sequence-to-Sequence (seq2seq) QG models are models which are build out of RNNs to convert sequences from one domain to another. The base of this architecture includes an Encoder and an Decoder (Chollet, 2017). The components of the seq2seq models which are placed between the Encoder and the Decoder consist of bucketing, search and attention (Kriangchaivech & Wangperawong, 2019).

Weights can be set that tell the Encoder that the sentences should be encoded as vectors. By adding those weights, the Encoder can be told on which words he should set his focus on. With all this information, the Decoder analyses the input sentence selectively and can then reproduce the input into an meaningful output sequence (Chollet, 2017).

The highest probability word occurrence can be achieved by beam search. This so called “beam search” is based on the occurrence of previous words in the sentence. Bucketing is a mechanism which makes a vary of the length of the sequences possible. It is depending on how the bucket size is designed in the first place. For correctly predicting the next word, the Decoder gets rewarded. If the Decoders makes incorrect predictions, the mechanism gets penalized (Kriangchaivech & Wangperawong, 2019).

Some state-of-the-art seq2seq models are: Long Short-Term Memory (LSTM), Recurrent Neural Networks (RNNs) and Gated Recurrent Unit (GRU) (Chollet, 2017). More information on these models can be found in Chapter *4.4 Classical/Traditional NLP Networks*.

5.3.2 Transformer based Models

Transformer based models show great possibilities for generating questions with minimal human invention by capturing a greater semantic representation of the given context. The benefits of its parallelizable architecture and no need of recurrence (because of its multiple attention heads) led Transformers to be better suited for dealing with larger text

corpus and little training data. Since the availability of German Question Generation datasets is limited, Transformers can show great benefits in such tasks (Kriangchaivech & Wangperawong, 2019).

6 Discussion of recent findings

Natural Language Processing is a very broad field in data science which still needs a lot of research especially for applications of Question Generation. Although conventional methods for generating sentences have delivered quite good results in past applications, there are some limitations that are problematic for larger or more complex input data and tasks. Examples of those limitations are the use of very domain specific words and the lack of understanding the semantics of the given input. Especially for specific tasks and applications with large textual input, such as generating questions from larger documents, it is very important to understand more of the surrounding semantics of the given input text. The use of newer models such as Transformers can offer a relief. One great advantage is that Transformers are license-free libraries which can be easily accessed online. Furthermore, various pretrained models have been published in the recent years. These Models have often been pretrained straightforward for one diverse task. Besides of that, they can easily be adapted for new, different downstream tasks by finetuning pretrained models. Also, combinations of different architectures, modules, and libraries should be considered for generating questions.

Which this knowledge earned, a framework for generating questions from German text is implemented within this paper. Two Transformer architectures are considered and some freely available models and libraries have been used. Further information can be found in the next chapter.

7 Experimental Setup

7.1 Definition and requirements

Generating Questions from text has been a popular research topic in recent years. Even though many of the recent application work well on phrase-like texts and/or single sentences, generating questions from larger textual documents without the need of additional human input are less common. Especially systems for German are rare.

The goal of this thesis is to invent a Question Generation Framework which makes it possible to generate open-ended question of larger German texts without the need of additional human knowledge. The generated questions in this paper are open-ended questions with respect to the given input sequence and current keyword of the generated keyword-list.

7.2 Dataset and available Models

7.2.1 Used BERT Transformer

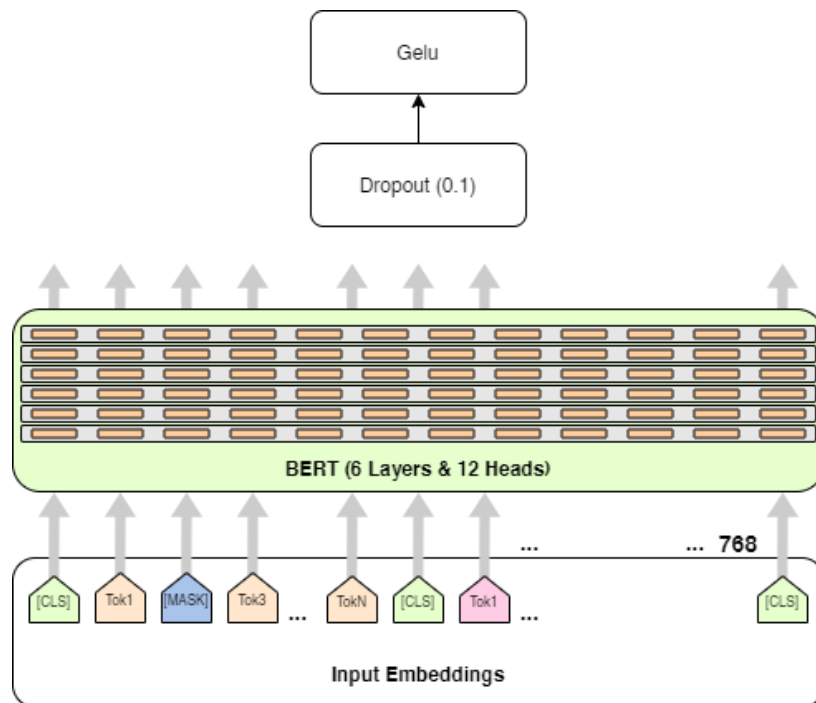


Figure 7: Architecture of used BERT (made by author).

To extract the most important sentences and keywords from the input text, a case-sensitive and distilled version of the multilingual BERT (distilbert-base-multilingual-cased) Transformer¹⁹ from Huggingface.co was applied. The cased DistilBERT base multilingual model is a faster version of the multilingual BERT-base model²⁰. It is trained on 104 languages by using Wikipedia as corpus. As illustrated in *Figure 7: Architecture of used BERT* (made by author). it consists of 6 layers, 768 dimensions, 12 heads and 134M parameters in total. This Transformer uses a dropout of 10% and a Gaussian Error Linear Unit (GELU) (Hendrycks & Gimpel, 2020) as activation function.

7.2.2 Used T5 Transformer

For the German Question Generation task proposed in this paper, the model that has been used for tokenization and generating questions is a modified version of Googles T5-small²¹ model. The proposed model was invented by Suraj Patil²² and is freely available online on Huggingface.co. It is a multi-task T5-small model which has already been trained for English QA and QG. Furthermore, to make this model applicable for German tasks, it got finetuned on the German XQuAD dataset. This dataset was released by (Artetxe et al., 2020) and is currently available for the public at Github.com²³ and Huggingface.co²⁴.

The proposed dataset itself consists of 240 paragraphs and 1190 question-answer pairs which have been extracted from the earlier published SQuADv1.1²⁵ dataset. Since SQuAD was invented for English language, the chosen data that has been used for XQuAD has been professionally translated into 10 different languages including German, Spanish, Russian, Greek, Arabic, Turkish, Vietnamese, Chinese, Thai and Hindi. An example of the German version of the XQuAD dataset is shown in *Figure 8: Example of the German XQuAD Dataset*

¹⁹ <https://huggingface.co/distilbert-base-multilingual-cased>, retrieved on 17.05.2021.

²⁰ <https://huggingface.co/bert-base-multilingual-cased>, retrieved on 26.06.2021.

²¹ <https://huggingface.co/t5-small>, retrieved on 17.05.2021.

²² <https://huggingface.co/valhalla/t5-small-qg-hl>, retrieved on 17.05.2021.

²³ <https://github.com/deepmind/xquad>, retrieved on 17.05.2021.

²⁴ <https://huggingface.co/datasets/xquad#xquadde>, retrieved on 17.05.2021.

²⁵ <https://deepai.org/dataset/squad1-1-dev>, retrieved on 17.05.2021.

id	context	question	answers
0 56beb4343aeaaa14008c925b	Die Verteidigung der Panthers gab nur 300 Punkte ab und belegte den sechsten Platz in der Liga, während sie die NFL mit 24 Interceptions in dieser Kategorie anführte und sich mit vier Pro Bowl-Selektionen rühmen konnte. Pro Bowl Defensive Tackle Kawann Short führte das Team mit 11 Sacks an, erzwang zudem drei Fumbles und erzielte zwei Fumble Recoverys. Mario Addison, ebenfalls Lineman, addierte 6½ Sacks hinzu. Die Panthers-Line präsentierte auch den erfahrenen Defensive End Jared Allen, einen 5-fachen Pro-Bowler, der mit 136 Sacks der aktive Anführer in der NFL-Kategorie Karriere-Sacks war, sowie den Defensive End Kony Ealy, der 5 Sacks in nur 9 Starts erzielte. Nach ihnen wurden zwei der drei Linebacker der Panthers ausgewählt, um im Pro Bowl zu spielen: Thomas Davis und Luke Kuechly. Davis erzielte 5½ Sacks, vier erzwungene Fumbles und vier Interceptions, während Kuechly das Team bei den Tackles anführte (118), zwei Fumbles erzwang und vier Pässe abfing. Carolinas Secondaries bestanden aus dem Pro Bowl-Safety Kurt Coleman, der das Team mit einem Karrierehoch von sieben Interceptions anführte und gleichzeitig 88 Tackles erzielen konnte, und Pro Bowl-Cornerback Josh Norman, der sich während der Saison zur Shutdown Corner entwickelte und vier Interceptions erzielte, von denen zwei zu Touchdowns für sein Team wurden.	Wie viele Punkte gab die Verteidigung der Panthers ab?	<pre>{ "answer_start": [38], "text": ["300"] }</pre>

Figure 8: Example of the German XQuAD Dataset (see Footnote 23).

7.2.3 Used Hardware

The hardware that has been used within the experimental setup consists of a Intel(R) Core(TM) i7-6900K Processor with a CPU up to 3.20GHz, 128 GB RAM, and a GTX 1080 Founders Edition graphic card. The used hard drive was a 1 TB Samsung EVO SSD.

7.3 Framework and Tools

The Question Generation system got implement using the PyTorch Framework. The PyTorch-Lightning²⁶ module, which is a lightweight wrapper for high performance AI tasks has been used for finetuning the T5 Transformer. This module proved to be very useful when it comes to working on limited hardware. By using the LightningModule, a full AI system can be implemented with just a few lines of code. For more details, please refer to the original PyTorch-Lightning documentation²⁷. Moreover, several additional modules/libraries and packages were implemented within this system. The most important tools are described in *section 7.5.1 T5 Finetuning Pipeline* and *section 7.6.1 Question Generation Pipeline*.

²⁶ <https://pytorch-lightning.readthedocs.io/en/latest/>, retrieved on 17.05.2021.

7.4 Conceptual Design

The Question Generation task presented in this paper is based on two main steps followed by several sub-steps. At the beginning, an intensive literature research was conducted to get an overview of the currently most relevant Transformer models for seq2seq applications. The conceptual design of the implemented architecture is illustrated in *Figure 9: Conceptual Design of the Framework*.

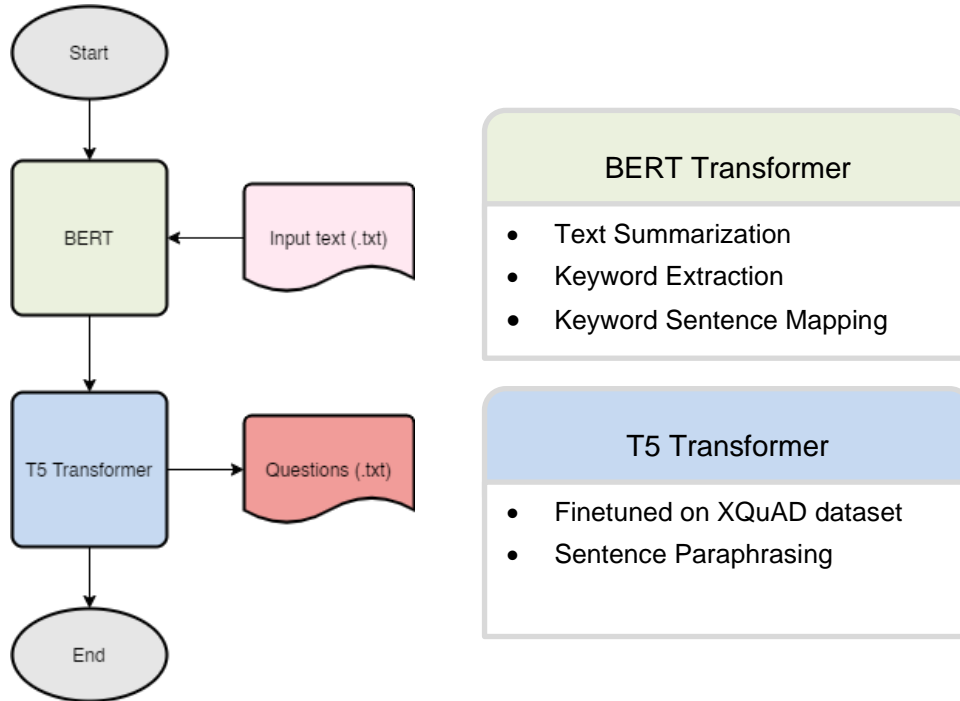


Figure 9: Conceptual Design of the Framework.

In this work, a distilled version of the multilingual BERT (see *Footnote 19*) has been chosen to summarize the provided input text (a plain .txt file). From this original text, the most important keywords were determined and stored in a separate list (Keyword Extraction). The sentences that contain the keywords from that list were first extracted from the summarized text and reunited afterwards (Keyword Sentence Mapping). This preselection of the main sentences was applied to ensure that questions are only formed from sentences considered as relevant.

For generating the questions, a pretrained T5 Transformer for English QA/QG tasks was used to rewire each of the sentences into question-like ones. For processing *German text*, this Transformer got finetuned on the German dataset XQuAD (see *Footnote 23*). For more detailed information, please refer to section 7.5.1 *T5 Finetuning Pipeline*.

7.5 Finetuning

Besides the pretraining of a model, the so-called "finetuning" is a good way to refine models for more specific downstream tasks. Especially in NLP, the risk of using a word with different meaning which often depends on the context is relatively high. Finetuning can be helpful for a better understanding of the semantic background of the input. Because of that, a more precise solutions for the use case can be found. The implemented pipeline is shown in *Figure 10: T5 Finetuning Pipeline.* Furthermore, in this section, the finetuning procedure of the T5 Transformer is described in more detail.

7.5.1 T5 Finetuning Pipeline

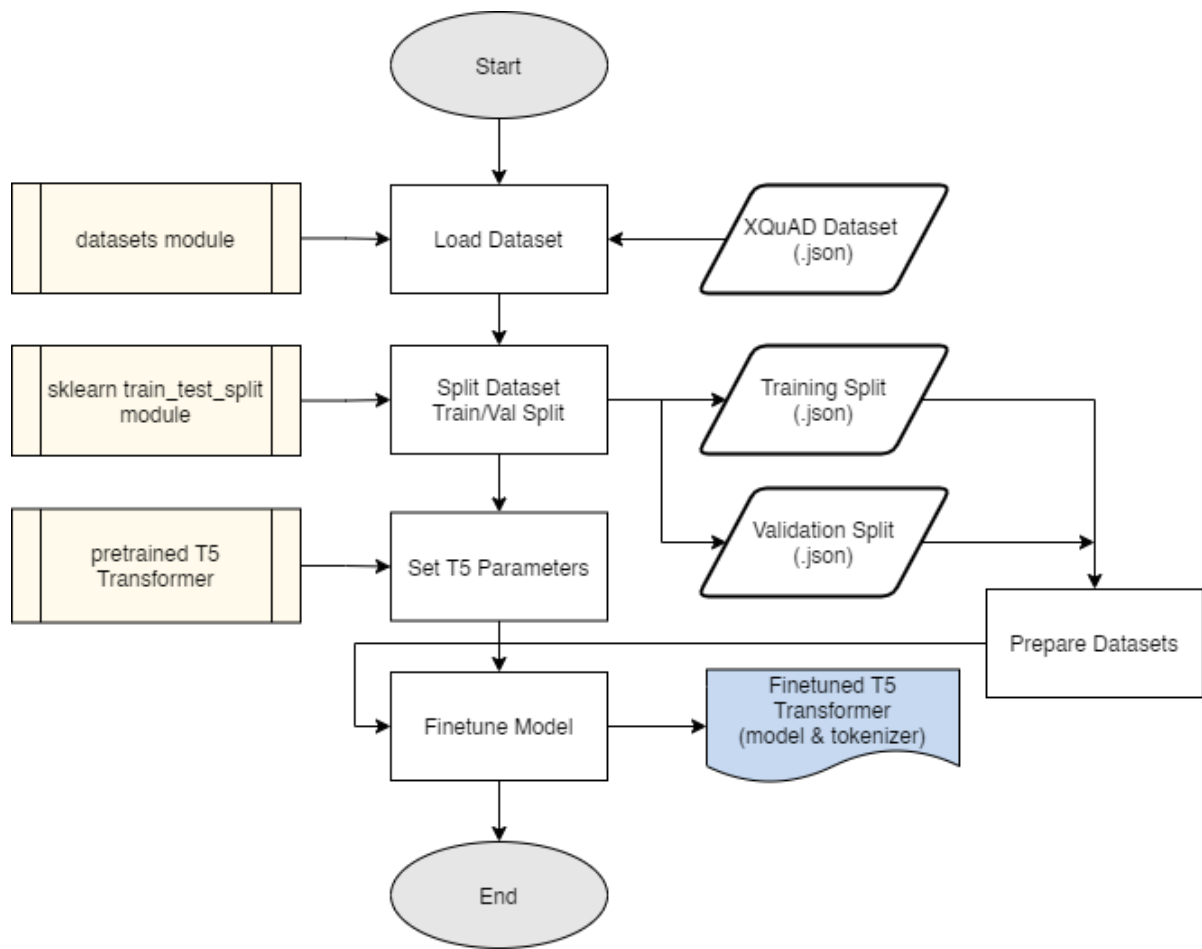


Figure 10: T5 Finetuning Pipeline.

1. **Load Dataset:** The proposed german XQuAD dataset (see Footnote 23) was downloaded from Huggingface.co. For further information and an example of the dataset please refer to chapter 7.2.2 Used T5 Transformer.

2. **Split Dataset Train/Val Split:** The downloaded XQuAD dataset was split into validation- and trainings-data (relation: 1:9).
 - Length of validation-data: 109 rows
 - Length of trainings-data: 981 rows
3. **Set T5 Parameters:** The parameters for finetuning the T5 Transformer were set using the Pytorch-Lightning (see *Footnote 26*) module.
4. **Dataset Preparation:** The dataset was compiled into the same input format as the SQuAD dataset of the pretrained model (see *Footnote 25*). A more in depth description can be found at section 7.5.2. *Dataset preparation and preprocessing for finetuning*.
5. **Finetune Model:** For finetuning the T5 Transformer the pretrained model of valhalla/t5-small-qg-hl (see *Footnote 22*) has been chosen. The parameters were adapted from the originally implemented Transformer. Nevertheless, some of them were set manually, including:
 - *Batch-size = 8 → number of processed samples before updating the model*
 - *Max_epochs = 15 → maximal amount of forward and backward passes of the entire training dataset*
 - *gpu = 1 → number of graphic processing units that should be used*
 - *early stopping → stops the model from training once the performance stops improving*

The model was trained for 14 epochs until the early stopping callback set in.

7.5.2 Dataset preparation and preprocessing for finetuning

For finetuning, the German QG/QA dataset XQuAD (see *Footnote 23*) has been used. This dataset already consists of the same syntax as the SQuAD dataset that has been used for pretraining the proposed model. Because of that, no certain preprocessing steps are needed. Since the available dataset is relatively little, it got split into a training- and validation-set in the relation of 1:9.

As mentioned in 7.2.2 *Used T5 Transformer*, the XQuAD-dataset consists of paragraphs and question-answer pairs just like the SQuAD dataset. Each row includes three columns: "context", "answer" and "question". For preparing the input data for the T5 Transformer, the XQuAD dataset has been formatted into the same input format as the SQuAD dataset that has been used for pretraining.

This customization was done in the implemented `QuestionGenerationDataset()`-class. Each row includes three columns: “context”, “answer” and “question”. As proposed at the original huggingface documentation of the pretrained T5 Transformer (see *Footnote 22*), the structures of the input and the target data were defined as followed:

Input data:

```
input_ = "context: %s <hl> answer: %s<hl> </s>" % (passage,  
answer)
```

Target:

```
target = "question: %s </s>" % (str(target))
```

The special token `</s>` got added to mark the end of the input and the target text. Moreover, the highlight-token “`<hl>`” was added to mark the key noun of the future question.

7.6 Question Generation

The pipeline of the actual Question Generation task is explained in more detail in this section. The main idea of the approach is to summarize the important passages of the input text, extracting the most important keywords, find the sentences that include the keyword and paraphrase those sentences into questions. The architecture of the application is illustrated in *Figure 11: Question Generation Pipeline*.

7.6.1 Question Generation Pipeline

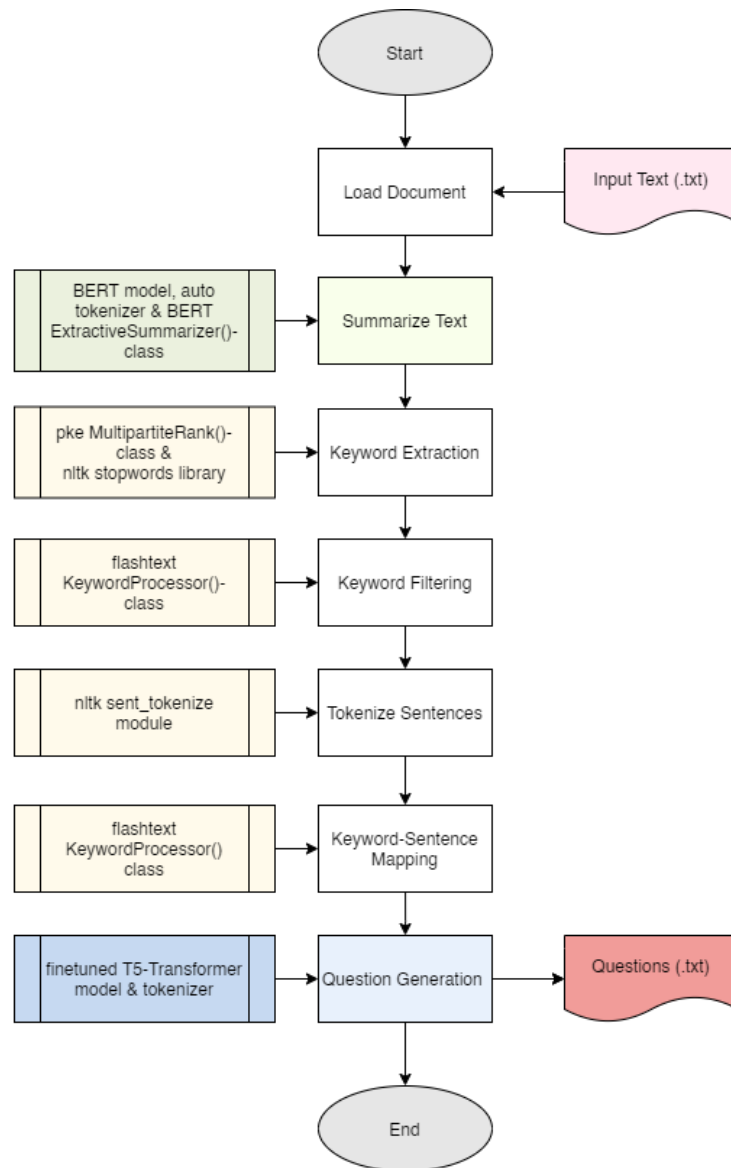


Figure 11: Question Generation Pipeline.

1. **Load Document:** The document proposed as an .txt file gets loaded into the system.
2. **Summarize Text:** For text summarization, the *BERT ExtractiveSummarizer()-class*²⁸ was used with the *distilbert-base-multilingual-BERT*²⁹ model from Huggingface.co.
3. **Keyword Extraction:** The most important keywords of the input file were filtered by using the unsupervised *pke MultipartiteRank()-class*³⁰. The implemented system only considers nouns as possible keywords. Moreover, by using the open source *nlTK.corpus* module³¹ a German stopwords list has been used for removing those nouns from the given input text.
4. **Keyword/Sentence mapping:**
Each sentence of the text has been tokenized using the *sent_tokenize()* method of the *nlTK.tokenize* module³².
Furthermore, each sentence which included one of the keyword-strings was determined by using the *KeywordProcessor()-class* of the *flashtext* module³³.
5. **Question Generation:**
A function called *prepare_inputs_for_qg_from_answers_hl()* got implemented for converting the sentences in the right input formats.
The main functionality of the *t2t_qg_with_t5()* method is to encode the input text, the input-ids and the attention-mask. Moreover, the parameters of the beams per sentence are set withing this function.

7.6.2 Dataset preparation and preprocessing for QG

For QG, the original sentences need to be preprocessed in a way that it fits the proposed input format of the pretrained T5 Transformer.

It was assumed that by adding the correct syntax into the system, the Transformer would not use the respective keywords to generate the questions. For that, the sentences

²⁸ <https://pypi.org/project/bert-extractive-summarizer/>, retrieved on 16.05.2021.

²⁹ <https://huggingface.co/distilbert-base-multilingual-cased>, retrieved on 16.05.2021.

³⁰ <https://boudinfl.github.io/pke/build/html/unsupervised.html>, retrieved on 16.05.2021.

³¹ <https://www.nltk.org/api/nltk.corpus.html>, retrieved on 16.05.2021.

³² <https://www.nltk.org/api/nltk.tokenize.html#module-nltk.tokenize>, retrieved on 16.05.2021.

³³ https://flashtext.readthedocs.io/en/latest/keyword_processor.html, retrieved on 16.05.2021.

should be prepared and added as mentioned in the official documentation of the T5 Transformer as shown in *Figure 12: T5 Input Documentation T5* (Patil, 2020).

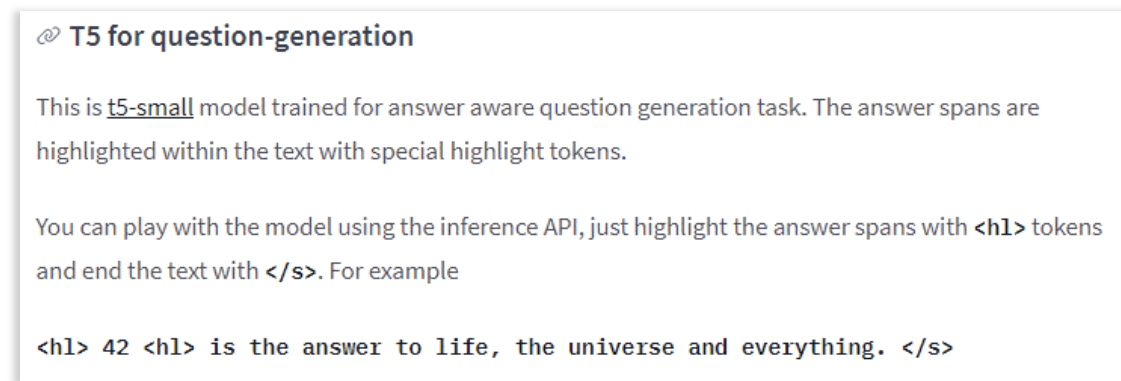


Figure 12: T5 Input Documentation T5 (Patil, 2020).

Unfortunately, for some (yet) unknown reason, this has not actually worked.

Because of that, different input formats were tested, but none of them delivered the expected output. In *Table 1: Chosen Experiments*, a brief overview of the most important experiments is given. For getting an insight into the other experiments, please refer to the official Github repository of the author³⁴.

³⁴ https://github.com/JaquJaqu/question_generation_de, retrieved on 08.07.2021.

Table 1: Chosen Experiments.

Approach Nr.2)	Approach Nr. 3)
<p>INPUT:</p> <p>Kurze Unterbrechungen sind normal, die <hl> Igel <hl> erwachen, bleiben im Nest und schlafen bald weiter.</p> <p>Question_Choices</p> <p>1: die normal sind, bleiben im Nest und schlafen bald weiter.</p> <p>2: die normal sind, bleiben im Nest</p>	<p>INPUT:</p> <p>context: Kurze Unterbrechungen sind normal, die <hl> Igel <hl> erwachen, bleiben im Nest und schlafen bald weiter.<hl>answer: Igel<hl></p> <p>Question_Choices</p> <p>1: Was bleiben im Nest und schlafen bald?</p> <p>2: Was bleibt im Nest und schlafen bald?</p>
Approach Nr. 4)	Approach Nr. 6)
<p>INPUT: context: Kurze Unterbrechungen sind normal, die <hl>Igel<hl> erwachen, bleiben im Nest und schlafen bald weiter.<hl>answer: Igel<hl></p> <p>Question_Choices</p> <p>1: Was bleiben im Nest und schlafen bald?</p> <p>2: Was bleibt im Nest und schlafen bald?</p>	<p>INPUT: generate question: Kurze Unterbrechungen sind normal, die Igel erwachen, bleiben im Nest und schlafen bald weiter.</p> <p>Question_Choices</p> <p>1: happens when Igel erwachen?</p> <p>2: happens when Igel erwacht?</p>

Interestingly, this experiments showed that the Transformer needs to have the prefix „answers“ with the <hl> tokens attached in order to produce questions-like sentences at all. Another interesting thing is, that experiments with the prefix „generate question:“ produced a question-like mixture of English- and German words. Even though the pretrained model was finetuned on generating only English sentences when the prefix “generate question” is used. In this case, no <hl> token is needed for the question-like syntax. Nevertheless, for finetuning the model for generating German questions, the prefix “context:” was added instead of “generate question”. Experiments with the “context”-prefix delivered the expected output for generating German questions. More information about the input format can be found in section 7.5.2 *Dataset preparation and preprocessing for finetuning*. Because of that, for evaluating the final T5 model, the experimental setup of approach number 4 has been chosen.

8 Evaluation and Results

8.1 Technique and Metrics

A model that is designed to generate questions is difficult to evaluate since the semantics and syntax of a question can differ a lot by still producing an valid/correct outcome. Amidei et al., 2018 is an excellent resource for getting an overview of the potential metrics and techniques that can be used by collecting approaches of recent systems that have taken advantage of methods for such tasks. For the evaluation of the system presented in this thesis, human assessment was chosen. Recent applications using human investigation mostly focused on evaluating specified criteria like the quality of a generated question. Other examples are its fluency, its grammatical and semantic correctness, and the relevance of the questions. But also, numerical scales and classification has been used for such assessments.

8.2 Input Texts

To investigate how the system behaves with textual inputs of different semantic complexity and both simple and subject-specific jargon, questions from text of three different levels of difficulty were generated.

The texts that have been used for evaluating the system were categorised into 3 classes: easy, moderate, and difficult texts. Each of these classes contains texts from 2 different resources. Those texts were manually converted by copy/pasting the source sentences into a plain .txt file.

Easy Texts

Text 1: Seven primary school texts about wildlife from www.grundschuleundbasteln.de³⁵ were merged into one .txt-file

Text 2: Blog entry for children about photosynthesis from kindersache.de³⁶

Moderate Texts

³⁵ <https://www.grundschuleundbasteln.de/2019/10/02/waldtiere-sachtexte/>, retrieved on 19.05.2021

³⁶ <https://www.kindersache.de/bereiche/wissen/natur-und-mensch/was-ist-photosynthese>, retrieved on 19.05.2021

Text 1: Article about herbs from www.wienerzeitung.at³⁷

Text 2: Article about African elephants from www.science.orf.at³⁸

Difficult Texts

Text 1: Lexica entry about thermodynamics from www.chemie.de³⁹

Text 2: Chapter 3.2 Recycling from the book “Recycling – ein Mittel zu welchem Zweck?” from Phillipp Schäfer (Schäfer, 2021)

³⁷ https://www.wienerzeitung.at/nachrichten/reflexionen/vermessungen/25696_Frische-Kraeuter-am-Balkon.html, retrieved on 19.05.2021

³⁸ <https://science.orf.at/stories/3205570/>, retrieved on 19.05.2021

³⁹ <https://www.chemie.de/lexikon/Thermodynamik.html>, retrieved on 19.05.2021

8.3 Evaluation Setup

For evaluation, 15 sentences of each difficulty level which have been considered as relevant by the algorithm where manually chosen. The texts were taken one after the other. Duplicates were not considered. If text 1 did not consist of 15 relevant sentences because the input text was too short, text 2 was used as a supplement for the missing sentences to provide the same number of sentences/questions for each difficulty level. For each of the sentences, five possible questions were generated.

A questionnaire was prepared using the online survey tool www.Limesurvey.org.⁴⁰ In that survey, the participants had to choose if the generated questions make sense, are grammatically correct, are related to the input and/or if the questions can be answered by referring to the given input sentences. A concrete example is shown in *Figure 13: Difficulty Level "easy" - Question Nr. 12.*

12.

Ursprünglicher Satz

Der Prozess der Photosynthese braucht Sonnenlicht, Kohlenstoffdioxid und Wasser.

	Die Frage macht Sinn	Die Frage ist grammatikalisch korrekt	Die Frage ist relevant	Die Frage bezieht sich auf den ursprünglichen Satz	Die Frage kann mit dem ursprünglichen Satz beantwortet werden
1. Was braucht Sonnenlicht, Kohlenstoffdioxid und Wasser?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
2. Welche Art von Prozess benötigt Sonnenlicht, Kohlenstoffdioxid und Wasser?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
3. Welche Art von Prozess braucht Sonnenlicht, Kohlenstoffdioxid und Wasser?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
4. Was benötigt Sonnenlicht, Kohlenstoffdioxid und Wasser?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
5. Was brauchen Sonnenlicht, Kohlenstoffdioxid und Wasser?	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

Figure 13: Difficulty Level "easy" - Question Nr. 12.

The questionnaire was spread on several mediums and social media like WhatsApp and Facebook.

⁴⁰ <https://www.limesurvey.org/de/>, retrieved on 18.05.2021.

8.4 Participants

Unfortunately, only 12 of the 92 participants completed the survey. The results of surveys that were not completely fulfilled were discarded. Most of the people who fulfilled the survey had a general higher education entrance qualification and stated that they have a very solid command of the German language. Many of the participants complained by personal feedback that the tasks of the questionnaire were too difficult, too confusing and that it would have taken too long to answer the questions. Because of that, they did not complete the survey. *Figure 14: General information of the participants* provides a graphical overview of all the 12 participants that have fulfilled the survey.

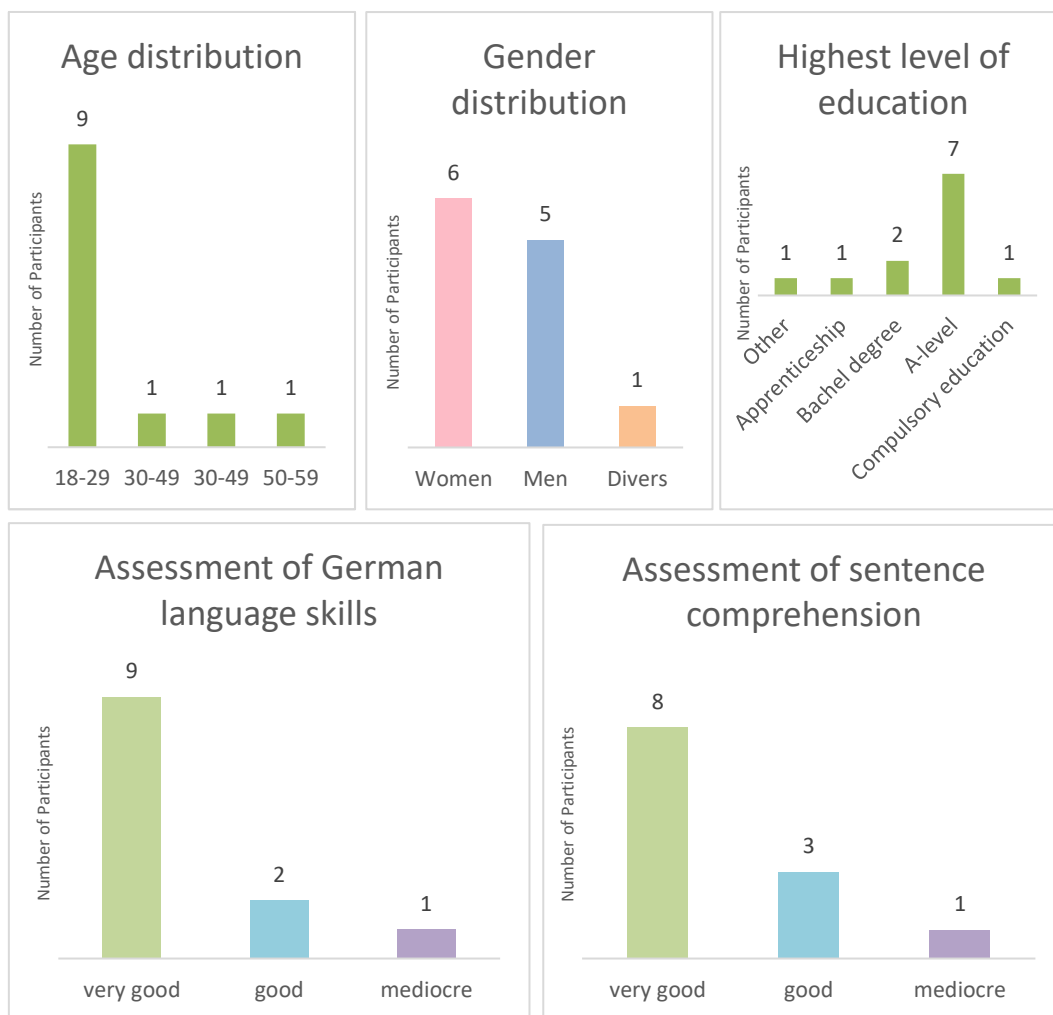


Figure 14: General information of the participants.

Since the number of participants was relatively small, the results are therefore more a matter of hypotheses than a statistically relevant outcome. In terms of that, the paper of

Amidei et al., 2018 also refers to the number of participants/annotators in the mentioned papers. It is said that within those papers the number of annotators varies from a minimum of 1 to a maximum of 364 persons. Comparing these different approaches, in average, 4 annotators have been used for evaluating such systems. Even though, the survey does not provide statistically relevant results and the participants were no trained annotators, it still offers a general overview over the invented system.

8.5 Results

The results were assessed in total with the texts of all difficulty levels at once, but also separately to the individual difficulty levels. This serves to make some assumptions on what extent the quality of the questions depends on the complexity of the input sentence. The generated questions and the answer choices were annotated by the survey tool as followed:

Answer Texts	
A001	Die Frage macht Sinn (engl.: „The question makes sense“)
A002	Die Frage ist grammatikalisch korrekt (engl.: „The question is grammatically correct“)
A003	Die Frage ist relevant (engl.: „The question is relevant“)
A004	Die Frage bezieht sich auf den ursprünglichen Satz (engl.: „The question refers to the original sentence“)
A005	Die Frage kann mit dem ursprünglichen Satz beantwortet werden (engl.: „The question can be answered with the original sentence“)
Generated Questions	
SQ001	Question Nr.1
SQ002	Question Nr.2
SQ003	Question Nr.3
SQ004	Question Nr.4
SQ005	Question Nr.5

Table 2: Annotation Limesurvey.

Two procedures that have been chosen to assess the outcome are shown in *Figure 15: Results Answer-Percentages*. First, the occurrence of each answer could be determined for each input sentence and each question generated from it. This was used to find out how often the respective answer options have been chosen.

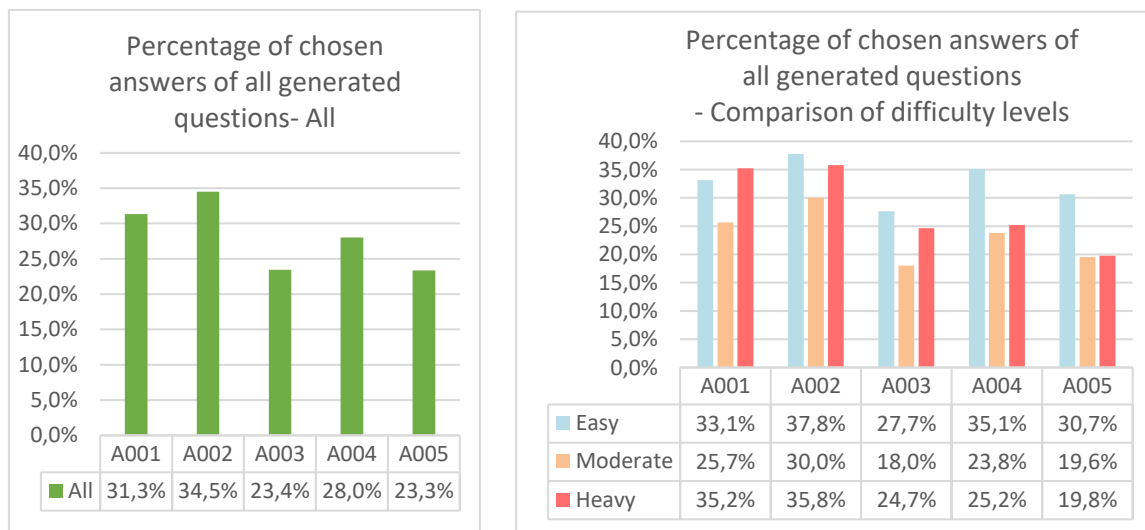


Figure 15: Results Answer-Percentages.

According to the graphs, around 32% of the outputs were senseful and correctly written questions. About 23% can be considered as relevant or can be answered with the source sentence. 28% of the generated questions refer to the given input sentences. Moreover, it seems that the questions generated from the “Easy” texts provide continuously better results than the more complex ones. Nevertheless, the relevance of the generated questions tends to be relatively low.

Since the goal of the thesis was to generate meaningful questions, the second procedure was invented to earn some knowledge about which of the 5 generated sub-questions make the most sense. These findings are illustrated in *Figure 16: Results Sense of Questions*.

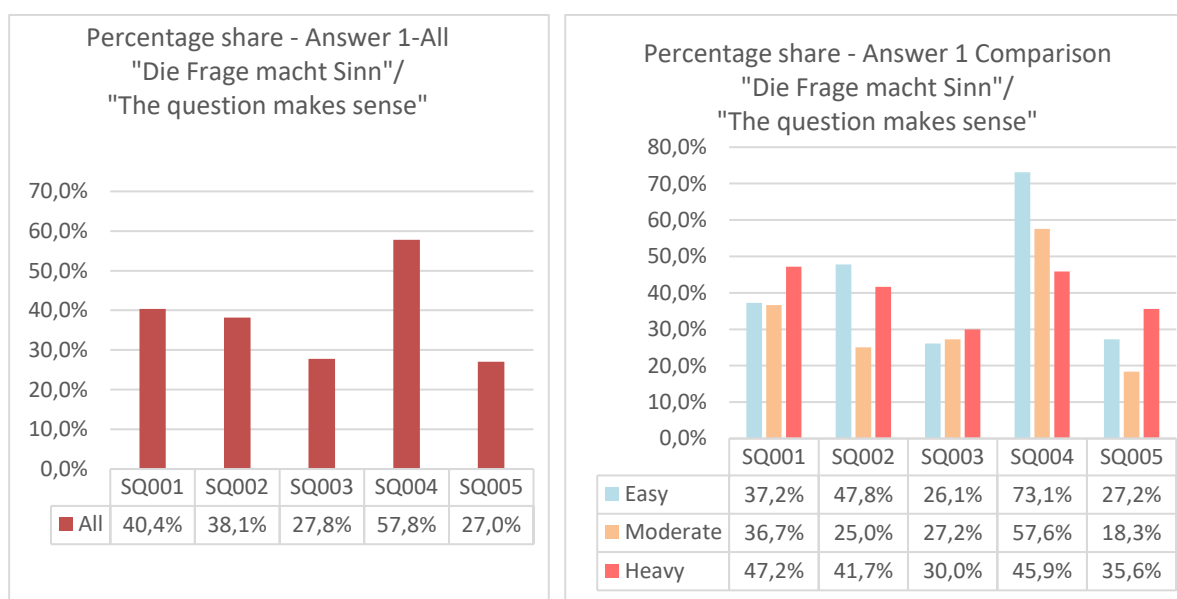


Figure 16: Results Sense of Questions.

The graphs show that in average, the fourth generated question for each sentence performs best when it comes to its semantics. Comparing the outcomes of each difficulty level, the easy texts give the best results in average. Questions generated from the easy text stand out with a score of around 73% considering the fourth generated question of the list.

9 Discussion of the Experiment

When skimming the generated questions manually, it is particularly noticeable that the variations of each question for each input sentence often only differ in minor details. Interestingly, the system seems to deliver relatively useful results when generating the 4th question.

Unfortunately, due to the lack of participants involved in the evaluation process, it is not feasible to state any statistically relevant conclusions. Nevertheless, it can be observed that the system delivers the best results with syntactically and semantically relatively simple texts. However, the results also show that it was more difficult for the system to generate relevant questions from the moderate texts than from the difficult ones, which is a contradiction to the first assumption. In this respect, when skimming the selected input sentences of the system, it is noticeable that some of the automatically selected sentences by using the BERT Transformer the sentences chosen from the moderate text seem to be somewhat taken out of context. For example, one of the sentences contains parts of the headline and another consists only individual keywords instead of a senseful sentence. To obtain better results here in future settings, attempting to make the summarizing of the texts more accurate could be beneficial. For that, other BERT Transformer models could be used. Unlike conventional methods, Transformer models can also be implemented in a way that they can understand the context of the surrounding sentences. This property could be implemented more effectively. Another attempt could be to use a T5 Transformer for the whole QG pipeline. This approach, however, requires further state-of-the-art research on German/multilingual models and/or relevant datasets. The full code of the experiments as well as the documentation can be found at the authors personal Github-repository (*see Footnote 34*).

10 Conclusion and Future Work

The research has shown that both, the traditional methods as well as the newer Transformer based models provide a good basis for generating questions. The biggest advantage of the Transformers compared to the traditional methods is that they can be adapted much more versatile for different tasks and languages. Often only a relatively small amount of data is necessary to adapt such a model to the respective task. A disadvantage is, however, that these models often require long training times and a relatively large amount of disk space. In this respect, it is positive to note that there are already some pretrained Transformers publicly available that can potentially be used to generate questions and/or can be used for sub-tasks within the whole pipeline (e.g., text summarization). In this work, the aggregation of the most meaningful texts as well as paraphrasing the chosen sentences were accomplished by using Transformers. A multilingual distilled BERT model was used for summarization and a finetuned T5 model for paraphrasing question-like sentences. Multilingual and/or german BERT-, T5- and GTP-Transformers are currently very popular in text generation. Unfortunately, only one German dataset was found during the research which can be used for finetuning the T5 model to accomplish to generate German questions. This dataset is the German version of the dataset XQuAD (see *Footnote 23*).

Although the aim of the work - to automatically generate German questions from texts - has been achieved, the results of the evaluation show that the quality of the questions is not yet adequate. The current application still needs human intervention to select high-quality questions.

Even though the grammatical correctness of the generated questions seems to be sufficient, it still needs human corrections for the most part. Considering the surrounding sentences of the input text might help to deliver a better outcome. By that, the system might have a better understanding of the semantics of the given input sentences. Another problem that must be considered when generating questions is that this task is difficult for questions with a very specific technical jargon. The problem is that the available pretrained T5 models simply do not know many of those terms. However, such words are particularly interesting for the generation of meaningful and more qualitative questions. Additionally, more focus on the keyword selection and the subsequent sentence mapping should be given. Other techniques for ranking the words could be applied to select the main keywords from the input text.

Since it was not possible to only generate questions that do not contain the respective keyword within the scope of this work, more investigation is needed here. If this could be

achieved, the system would be much more flexible. Other question types could be generated. For example, Multiple Choice Questions could be easily generated by adding additional Distractors as possible choices besides the original keywords.

Experiments with other multilingual and/or German T5 Transformers should be done in future development. The greatest restriction of the idea of using a pretrained model and finetuning it on German tasks is, that unfortunately, the amount of proper German datasets especially for QG/QA is relatively rare. Therefore, it makes sense to use some resources for the development of new, larger German datasets followed by the attempt to finetune an already pretrained T5 Transformer with it. Since the mentioned approaches are carried out in a supervised manner, an unsupervised approach for teaching the Transformer a larger number of German texts and finetuning it afterwards on German question-/answer- pairs should also be considered. This attempt could bring great benefit to make the model more domain independent.

References

- Amidei, J., Piwek, P., & Willis, A. (2018). Evaluation methodologies in Automatic Question Generation 2013-2018. In *Proceedings of The 11th International Natural Language Generation Conference* (pp. 307–317).
- Anand, A. (2020, July 21). *Anatomy of Language Models In NLP*. DEV Community. <https://dev.to/amananandrai/language-models-in-nlp-21jn>
- Artetxe, M., Ruder, S., & Yogatama, D. (2020). On the Cross-lingual Transferability of Monolingual Representations. *ArXiv:1910.11856* [Cs]. <http://arxiv.org/abs/1910.11856>
- Chan, Y.-H., & Fan, Y.-C. (2019). A Recurrent BERT-based Model for Question Generation. *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, 154–162. <https://doi.org/10.18653/v1/D19-5821>
- Chollet, F. (2017, September 29). *A ten-minute introduction to sequence-to-sequence learning in Keras* [Blog]. The Keras Blog. <https://blog.keras.io/a-ten-minute-introduction-to-sequence-to-sequence-learning-in-keras.html>
- Dong, L., Yang, N., Wang, W., Wei, F., Liu, X., Wang, Y., Gao, J., Zhou, M., & Hon, H.-W. (2019). Unified Language Model Pre-training for Natural Language Understanding and Generation. *ArXiv:1905.03197* [Cs]. <http://arxiv.org/abs/1905.03197>
- Du, X., Shao, J., & Cardie, C. (2017). Learning to Ask: Neural Question Generation for Reading Comprehension. *ArXiv:1705.00106* [Cs]. <http://arxiv.org/abs/1705.00106>
- Goldberg, Y. (2015). A Primer on Neural Network Models for Natural Language Processing. *ArXiv:1510.00726* [Cs]. <http://arxiv.org/abs/1510.00726>

- Hendrycks, D., & Gimpel, K. (2020). Gaussian Error Linear Units (GELUs). *ArXiv:1606.08415 [Cs]*. <http://arxiv.org/abs/1606.08415>
- Hochreiter, S., & Schmidhuber, J. (1997). Long Short-Term Memory. *Neural Computation*, 9(8), 1735–1780. <https://doi.org/10.1162/neco.1997.9.8.1735>
- Horev, R. (2018, November 17). *BERT Explained: State of the art language model for NLP*. Medium. <https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270>
- Hutchison, D., Kanade, T., Kittler, J., Kleinberg, J. M., Mattern, F., Mitchell, J. C., Naor, M., Nierstrasz, O., Pandu Rangan, C., Steffen, B., Sudan, M., Terzopoulos, D., Tygar, D., Vardi, M. Y., Weikum, G., Aldabe, I., & Maritxalar, M. (2010). Automatic Distractor Generation for Domain Specific Texts. In H. Loftsson, E. Rögnvaldsson, & S. Helgadóttir (Eds.), *Advances in Natural Language Processing* (Vol. 6233, pp. 27–38). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-14770-8_5
- Jain, A., Department of Computer Engineering, SVKMs NMIMS MPSTME Shirpur, Maharashtra, India, Kulkarni, G., Department of Computer Engineering, SVKMs NMIMS MPSTME Shirpur, Maharashtra, India, Shah, V., & Department of Computer Engineering, SVKMs NMIMS MPSTME Shirpur, Maharashtra, India. (2018). Natural Language Processing. *International Journal of Computer Sciences and Engineering*, 6(1), 161–167. <https://doi.org/10.26438/ijcse/v6i1.161167>
- Jing, K., & Xu, J. (2019). A Survey on Neural Network Language Models. *ArXiv:1906.03591 [Cs]*. <http://arxiv.org/abs/1906.03591>
- Kapadia, S. (2019, September 5). *Topic Modeling in Python: Latent Dirichlet Allocation (LDA)*. Medium. <https://towardsdatascience.com/end-to-end-topic-modeling-in-python-latent-dirichlet-allocation-lda-35ce4ed6b3e0>

- Kriangchaivech, K., & Wangperawong, A. (2019). Question Generation by Transformers. *ArXiv:1909.05017 [Cs]*. <http://arxiv.org/abs/1909.05017>
- Kumawat, D. C. (2020, January 10). *Introduction to Term Frequency—Inverse Document Frequency(TF-IDF) in Natural Language Processing....* Medium. <https://medium.com/@sdinesh718/introduction-to-term-frequency-inverse-document-frequency-tf-idf-in-natural-language-processing-3ec503ddbdbab>
- Kurdi, G., Leo, J., Parsia, B., Sattler, U., & Al-Emari, S. (2020). A Systematic Review of Automatic Question Generation for Educational Purposes. *International Journal of Artificial Intelligence in Education*, 30(1), 121–204. <https://doi.org/10.1007/s40593-019-00186-y>
- Lavine, B. K., & Blank, T. R. (2009). Feed-Forward Neural Networks. In *Comprehensive Chemometrics* (pp. 571–586). Elsevier. <https://doi.org/10.1016/B978-044452701-1.00026-0>
- Le, J. (2019, June 15). *Recurrent neural networks: The powerhouse of language modeling*. Built In. <https://builtin.com/data-science/recurrent-neural-networks-powerhouse-language-modeling>
- Lindberg, D. L. (2013). *Automatic question generation from text for self-directed learning* [Thesis) M.Sc., Simon Fraser University]. <http://summit.sfu.ca/item/12985>
- Lopez, L. E., Cruz, D. K., Cruz, J. C. B., & Cheng, C. (2020). Transformer-based End-to-End Question Generation. *ArXiv:2005.01107 [Cs]*. <http://arxiv.org/abs/2005.01107>
- Mayo, M. (2017, December). A General Approach to Preprocessing Text Data. *KDnuggets*. <https://www.kdnuggets.com/a-general-approach-to-preprocessing-text-data.html/>

- Nguyen, V. H., Nguyen, H. T., & Snasel, V. (2016). Text normalization for named entity recognition in Vietnamese tweets. *Computational Social Networks*, 3(1), 10. <https://doi.org/10.1186/s40649-016-0032-0>
- Oberoi, A. (2020, July 15). *What are Language Models in NLP?* <https://insights.daffodilsw.com/blog/what-are-language-models-in-nlp>
- Padmanabhan, A. (2020, May 23). *Question Generation*. Devopedia. <https://devopedia.org/question-generation>
- Patil, S. (2020). *Valhalla/t5-small-qa-qg-hl*. Hugging Face. Huggingface. <https://huggingface.co/valhalla/t5-small-qa-qg-hl>
- Prateek, J. (2019, June 19). Transformers In NLP | State-Of-The-Art-Models. *Analytics Vidhya*. <https://www.analyticsvidhya.com/blog/2019/06/understanding-transformers-nlp-state-of-the-art-models/>
- Raffel, C., Shazzer, C., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P. J., Mankin, K., Fiedel, N., & Dinculescu, M. (2020, February 24). Exploring Transfer Learning with T5: The Text-To-Text Transfer Transformer. *Google AI Blog*. <http://ai.googleblog.com/2020/02/exploring-transfer-learning-with-t5.html>
- Rogers, A., Kovaleva, O., & Rumshisky, A. (2020). A Primer in BERTology: What we know about how BERT works. *ArXiv:2002.12327 [Cs]*. <http://arxiv.org/abs/2002.12327>
- Schäfer, P. (2021). *Recycling - ein Mittel zu welchem Zweck? Modellbasierte Ermittlung der energetischen Aufwände des Metallrecyclings für einen empirischen Vergleich mit der Primärgewinnung*.
- Sherstinsky, A. (2020). Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D: Nonlinear Phenomena*, 404, 132306. <https://doi.org/10.1016/j.physd.2019.132306>

- TIET, Nohria, A., & Kaur, H. (2018). Evaluation of Parsing Techniques in Natural Language Processing. *International Journal of Computer Trends and Technology*, 60(1), 31–34. <https://doi.org/10.14445/22312803/IJCTT-V60P104>
- Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2017). Attention Is All You Need. *ArXiv:1706.03762 [Cs]*. <http://arxiv.org/abs/1706.03762>
- Yao, X., Bouma, G., & Zhang, Y. (2012). Semantics-based Question Generation and Implementation. *Dialogue & Discourse*, 3(2), 11–42. <https://doi.org/10.5087/dad.2012.202>
- Zhou, Q., Yang, N., Wei, F., Tan, C., Bao, H., & Zhou, M. (2018). Neural Question Generation from Text: A Preliminary Study. In X. Huang, J. Jiang, D. Zhao, Y. Feng, & Y. Hong (Eds.), *Natural Language Processing and Chinese Computing* (Vol. 10619, pp. 662–671). Springer International Publishing. https://doi.org/10.1007/978-3-319-73618-1_56

List of figures

Figure 1: Example of category-pattern-template block (Lindberg, 2013).	9
Figure 2: Examples of template-based questions (Lindberg, 2013).....	9
Figure 3: Transformer Architecture (made by author, based on Neptune.ai).	15
Figure 4: BERT Fully-Connected Layer (made by author inspired by ai.googleblog.com).....	16
Figure 5: BERT Tokenization (made by author, based on towardsdatascience.com).	17
Figure 6: Architecture T5 Transformer (made by author, based on medium.com).	19
Figure 7: Architecture of used BERT (made by author).....	25
Figure 8: Example of the German XQuAD Dataset (see Footnote 23).	27
Figure 9: Conceptual Design of the Framework.	28
Figure 10: T5 Finetuning Pipeline.	29
Figure 11: Question Generation Pipeline.	32
Figure 12: T5 Input Documentation T5 (Patil, 2020).....	34
Figure 13: Difficulty Level "easy" - Question Nr. 12.	38
Figure 14: General information of the participants.....	39
Figure 15: Results Answer-Percentages.	41
Figure 16: Results Sense of Questions.	42

List of tables

Table 1: Chosen Experiments.....35

Table 2: Annotation Limesurvey.....41