



**Utrecht
University**

Knowledge Graph Expansion Using Question Answering By Leveraging Pre-Trained Language Models

MASTER'S THESIS

submitted in partial fulfillment of the requirements for the degree of

Master of Science

in

Applied Data Science

by

Hagen Aad Fock, B.Sc.

Student Number 1935569

to the Faculty of Science

at the Utrecht University

Examinor: Dr. Hakim AAA Qahtan

Advisor: Dr. Mel W. Chekol

Duygu Sezen Islakoğlu, M.Sc.

Utrecht, 1st July, 2022

Hagen Aad Fock

Hakim AAA Qahtan

Abstract

Knowledge Graphs are powerful and flexible data structures consisting of factual triples, such as (Barack Obama, isPresident, USA). Among other things, these factual triples might have a validity period, making a temporal component relevant. It is time-consuming to extend a knowledge graph to a temporal knowledge graph, turning the triple into a quadruple by adding temporal information.

The advances in the architecture of Language Models, which allow the unification of immense knowledge, open a possibility to utilise these models as temporal knowledge bases to automate the knowledge graph expansion eventually. In particular, there is an interest in evaluating the capability of current state-of-the-art models using temporal closed-book questions to find out if they have potential as temporal knowledge bases.

To get to the bottom of this interest, state-of-the-art text-to-text Pre-Trained Language Models capable of answering temporal closed-book questions were evaluated. This was done by converting the Temporal Knowledge Graph YAGO11k into different forms of temporal closed-book questions using the question generation pipeline developed in this thesis called TKGQuestionGenerator. Subsequently, the Text-To-Text Pre-Trained Language Models T0_3B and T0pp by BigScience were queried on the generated questions, and the results were evaluated.

The experiments revealed that the surveyed Pre-Trained Language Models using temporal closed-book questions lacked quality as a temporal knowledge base. Taking into account the omission of the fine-tuning to the desired questions, a rudimentary understanding of T0_3B and T0pp could nevertheless be determined, which scales by the size of the model and thus cannot be ruled out that more extensive and better tuned Pre-Trained Language Models in the future will have the quality to serve as a temporal knowledge base.

Contents

1	Introduction	1
2	Background	5
3	Related Work	9
4	Method	11
4.1	Framework	11
4.2	TKGQuestionGenerator	11
4.3	Text-To-Text Pre-Trained Language Models	14
4.4	Assessment of the Models' Answers	16
5	Data analysis	19
5.1	Chosen Temporal Knowledge Graph	19
5.2	Missing Data	20
5.3	Generated Questions	21
6	Empirical Evaluation	25
6.1	Evaluation	25
6.2	Limitations	32
7	Discussion, Ethics and Future Work	35
7.1	Discussion	35
7.2	Ethical Concerns	36
7.3	Future Work	37
8	Conclusion	39
	List of Figures	41
	List of Tables	41
	List of Code	42
	Acronyms	43

Bibliography	45
Appendix - Summary Grouped by Predicate	51
Appendix - Assessment of the Models' Answers Python Functions	57

Introduction

Knowledge Graph (KG) consists of many factual triples in the formats (Subject, Predicate, Object) or (Head, Relation, Tail) under the Resource Description Framework (RDF). A factual triple consists of a subject, an object and the relationship of how the subject relates to the object. An accumulation of many triples is a flexible data model with much expressive power to describe complex circumstances and relationships by integrating knowledge into an ontology and is capable of applying reasoner to derive new knowledge (Ji, S. Pan, Cambria, Marttinen, and Yu, 2020; Ehrlinger and Wöß, 2016). The human race is a never resting, uninterrupted engine of knowledge creation and many ideas have already been invented and expanded to store knowledge, and the KG is one of them. However, if a KG is continuously enriched with knowledge, facts represented by the triples can become ambiguous because facts can overlap from the statement. For instance, (Barack Obama, isPresident, USA) was correct during the period 2009-2017, but after that (Donald Trump, isPresident, USA) was correct during 2017-2021 and (Joe Biden, isPresident, USA) is currently correct since 2021. The validity time is one of many reasons why it is beneficial to expand a KG into a Temporal Knowledge Graph (TKG) with factual quadruples, which possesses time information, such as 2009-2017 in (Barack Obama, isPresident, USA, 2009-2017). KG expansion to a TKG is the process of adding time information and expanding KGs is time-consuming and costly, but there is a potential for automation which reduces the costs by leveraging Pre-trained Language Model (PLM).

A new architecture for Language Model (LM) called Transformers was proposed by Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin (2017), and they shifted the state-of-the-art architecture of Natural Language Processing (NLP) models because those achieved and surpassed state-of-the-art results in many downstream tasks. By making Recurrent Neural Network (RNN), Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) and Gated Recurrent Units (GRU) (Chung, Gülçehre, KyungHyun Cho, and Y. Bengio, 2014) superfluous it became possible to pre-train LMs

in parallel on massive corpora. Additionally, Transformers entirely rely on self-attention to compute representations of their input and output, allowing Self Supervised Learning (SSL) (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin, 2017). The two aspects combined, SSL and parallel training, allow those models to internalise massive corpora of unlabelled data, which holds enormous knowledge. An instance of such corpora is Wikipedia. The first model of its kind was BERT (Devlin, M. Chang, Kenton Lee, and Toutanova, 2019). BERT and its derivatives achieved the best performance of the most common NLP benchmarks (Devlin, M. Chang, Kenton Lee, and Toutanova, 2019; L. Dong, N. Yang, W. Wang, Wei, X. Liu, Y. Wang, J. Gao, M. Zhou, and Hon, 2019; Lan, Mingda Chen, Goodman, Gimpel, P. Sharma, and Soricut, 2019) due to transfer-learning the pre-trained knowledge towards downstream tasks by fine-tuning the models.

Since Raffel, Shazeer, Roberts, Katherine Lee, Narang, Matena, Y. Zhou, Li, and P. J. Liu (2019) proposed the PLM T5 which has a text-to-text framework it is also possible to query such models by closed-book questions. The text-to-text framework takes text as an input and produces new text as an output. Roberts, Raffel, and Shazeer (2020) showed that those models, given sufficient parameters, are able to memorise some factual knowledge within the parameter weights.

The main goal is to expand KGs based on their factual triples and a human intuitive way to retrieve certain information is by asking questions. Hence, we can use these models to answer questions, for instance, by having (Barack Obama, isPresident, USA) as a factual triple a natural approach would be to ask for 'When was Barack Obama president of USA?'. Such setup enforces the language model to answer questions based on knowledge that it internalised in its parameters during pre-training.

Although these PLM contain an incredible amount of knowledge, it is not yet certain whether they can act as a reliable knowledge base. Petroni, Rocktäschel, Lewis, Bakhtin, Y. Wu, Miller, and Riedel (2019) suggested that, for instance, the PLM BERT has the potential to be a knowledge base without fine-tuning. However, Pörner, Waltinger, and Schütze (2019) discovered that Petroni, Rocktäschel, Lewis, Bakhtin, Y. Wu, Miller, and Riedel (2019) must have drawn the wrong conclusions because their used data was distorted which made named-based reasoning possible and thus this the data easy to guess. Pörner, Waltinger, and Schütze (2019) concluded those PLMs are no knowledge bases yet. However, Roberts, Raffel, and Shazeer (2020) showed that those models achieve competitive results on open-domain question answering, also indicating the potential of being a knowledge base. Their arguments are mainly based on factual knowledge but not on temporal knowledge. There is yet little research that explores if those PLMs are capable of temporal knowledge or the ability to reason about time and thus if they have the potential to be leveraged as temporal knowledge bases. Dhingra, Cole, Eisenschlos, Gillick, Eisenstein, and Cohen (2021) evaluated the temporal knowledge of PLMs by asking them time dependent factual questions, such as an athlete's team or the name of the president which couples the answer with a validity time. For TKG expansion, the extraction of periods and times is desired and this has not yet been comprehensively

investigated.

The contribution of this study is to gain insight to which extend nowadays PLMs are capable to act as robust temporal knowledge bases to eventually use them as an automated method for KG expansion into a TKG using question answering. An intermediate step is the evaluation of the formulation of Temporal Closed-Book Question (TCBQ) to understand the performance fluctuations as well as evaluating the accuracy and the robustness of such models regarding temporal knowledge. Thus following research question were defined.

- How much does the TCBQ formulation affect the performance of PLMs?
- How robust are PLMs when given yes no questions with correct and incorrect years?
- What is the performance of PLMs regarding temporal closed book questions?

Within the scope of this thesis a pipeline called TKGQuestionGenerator (QG) was build to generate TCBQ based on KGs triples or TKGs quadruples. The generation based on quadruples returns beside the question the temporal information for evaluation reasons.

Background

Natural Language Processing (NLP) describes techniques and methods for machine processing of natural language. The goal is direct communication between humans and computers based on natural language (Otter, Medina, and Kalita, 2021).

Transformers were introduced in 2017 by a team at Google Brain (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin, 2017). They became the NLP models' architecture of choice to achieve the best results. Transformers are based on an encoder-decoder structure. Using self-attention, inputs can interact with each other, evaluating which part of the input should be paid attention to. Before transformers, most former state-of-the-art NLP systems relied on gated RNNs incorporating LSTM and GRUs, with added attention mechanisms (Hochreiter and Schmidhuber, 1997; Chung, Gülçehre, KyungHyun Cho, and Y. Bengio, 2014; Sutskever, Vinyals, and Le, 2014; Kyunghyun Cho, Merrienboer, Gülçehre, Bougares, Schwenk, and Y. Bengio, 2014). The vanishing gradient problem is the flaw why the sequential processing of tokens in RNNs cannot perform to its full potential and performance is compromised as a result. The dependency of token computations on results of previous token computations also makes it hard to parallelise computation on modern deep learning hardware because sequential processing requires the previous intermediate solutions and thus resulting in inefficient training of RNNs (Kolen and Kremer, 2001). These problems were solved by solely concentrating on attention mechanisms, as the title of (Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin, 2017) 'Attention is all you need' says, and by getting rid of RNN. Language translation is an adequate example to show the value of attention because context is crucial to assigning the meaning of a word in a sentence. For instance, an English-French translation system is taken, where the first word of the French output most likely depends heavily on the first few words of the English input. In a classical LSTM mechanism, only the state vector after processing the last English word can be accessed to produce the first word of the French output. According to theory, this vector contains any information regarding the entire sentence.

In practice, this information is insufficiently stored. However, this problem is solvable with self-attention, a variant of attention. Self-attention processes a sequence by replacing each element with a weighted average of the rest of the sequence, in other words giving the decoder access to the state vectors of each English input word. The vectors' attention weights can be learned that indicate the degree to which each English input state vector should be considered.

Self-supervised learning (SSL) is the typical form used when training transformers. The basic idea of SSL is to solve tasks that result only from the input. One possibility would be to cut out a part of the input and give the model the task of predicting the cut-out part. An example, the masked language model is a self-supervised task that attempts to predict the masked words in a sentence from the remaining words (Qiu, T. Sun, Y. Xu, Shao, N. Dai, and Huang, 2020). For instance, "How <masked> you?" is the masked sentence and the model should predicate which word most likely represents the <masked> word and in this case it would be "are". SSL consists of two distinct steps, pre-training and fine-tuning. In the first step, pre-training, network weights are initialised using large amounts of unstructured data. This process induces a general understanding of human language into the transformer model. The second step is a supervised fine-tuning of the model based on the desired downstream task. Typical pre-training and fine-tuning tasks include language modelling, next sentence prediction, question answering, reading comprehension, sentiment analysis and paraphrasing (A. Wang, Singh, Michael, Hill, Levy, and Bowman, 2018). SSL has arisen from the dilemma that there are only a relatively limited number of annotated records in relation to the enormous amount of unannotated text on the Internet. Using the prior knowledge and human language understanding the model has received from pre-training, it can be fine-tuned with small datasets for downstream tasks such as answering questions, resulting in a significant improvement in accuracy compared to training with these datasets from scratch¹. Pre-training has another advantage it can also be regarded as a regularisation that prevents the model from over-fitting (Erhan, Y. Bengio, Courville, Manzagol, Vincent, and S. Bengio, 2010).

Downstream task is used in the self-supervised learning context as a supervised task that leverages a pre-trained model. The downstream task is the actual problem the model should solve.

Transfer Learning is a research field in machine learning Machine Learning (ML) which focuses on accumulating knowledge acquired by solving a specific task and using the stored knowledge for a different but related task (L. Yang, Hanneke, and Carbonell, 2013; S. J. Pan and Q. Yang, 2010). When a model is first pre-trained on a large volume of data and then fine-tuned on a downstream task, it applies transfer learning and is broadly used for NLP tasks. In NLP are many variations of transfer learning, for instance, domain adaptation, cross-lingual learning and multi-task learning (Qiu, T. Sun, Y. Xu, Shao, N. Dai, and Huang, 2020).

¹Open Sourcing BERT: State-of-the-Art Pre-training for Natural Language Processing

Fine tuning is currently one of two ways to train a model. The second option is called feature extraction. By applying feature extraction, all parameters are frozen, and it is necessary to expose the internal layers as they encode the most transferable representations (Peters, Ruder, and Smith, 2019). Therefore, it needs complex task-specific architecture and is not as commonly used as fine-tuning (Qiu, T. Sun, Y. Xu, Shao, N. Dai, and Huang, 2020). Whereby by fine-tuning, the parameters are unfrozen, and the supervised tasks fine-tune the model. Fine-tuning is more general and convenient for many different downstream tasks than feature extraction.

Question Answering is the task of obtaining accurate answers to questions asked in natural language. Usually, a structured knowledge base is taken. Based on it, questions are asked to get answers from a question-answering system that has access to an unstructured collection of natural language documents. It is a typical downstream task of NLP models. Here, a distinction is made between answering questions in closed domains and answering questions in open domains. In the closed domain, domain-specific knowledge is retrieved, which can be formalised as ontologies. However, when answering open-domain questions, the questions can target almost anything based on general ontologies and world knowledge (Bouziane, Bouchiha, Doumi, and Malki, 2015; Zhu, Lei, C. Wang, Zheng, Poria, and Chua, 2021).

Closed-book question answering is a term used by Roberts, Raffel, and Shazeer (2020) to describe a particular circumstance of questioning NLP models. There are papers on question answering where the model has been explicitly fed with the desired information. For example, a suitable article is given as an extra input, and the answer to the question is found in the article. (Rajpurkar, J. Zhang, Lopyrev, and Liang, 2016; S. Zhang, X. Liu, J. Liu, J. Gao, Duh, and Durme, 2018; C. Clark, Kenton Lee, M. Chang, Kwiatkowski, Collins, and Toutanova, 2019) or an external source of knowledge has been made available to the model from which it can retrieve the desired data (D. Chen, Fisch, Weston, and Bordes, 2017). However, Roberts, Raffel, and Shazeer (2020) wanted to find out how much information is stored in the parameters of the respective models and have called this task closed-book question answering. Since the knowledge of the stored parameters is also queried within this work, the term closed-book question answering is also used here.

Knowledge Graphs (KG) have taken on a valuable role as a form of representation of structured human knowledge (X. L. Dong, Gabrilovich, Heitz, Horn, Lao, Murphy, Strohmman, S. Sun, and W. Zhang, 2014; Nickel, Murphy, Tresp, and Gabrilovich, 2015; Hogan, Blomqvist, Cochez, d’Amato, Melo, Gutierrez, Gayo, Kirrane, Neumaier, Polleres, Navigli, Ngomo, Rashid, Rula, Schmelzeisen, Sequeda, Staab, and Zimmermann, 2020). Facts are the basis of KGs. A fact consists of the directed relation of two entities with a semantic description. Entities can be real objects and abstract concepts. Each KG is made up of several factual triples. These factual triples are expressed in the format (Subject, Predicate, Object) or (Head, Relation, Tail) under the Resource Description Framework (RDF), for example (Cristiano Ronaldo, plays for, Real Madrid). The knowledge graph can be represented as a directed graph with nodes as entities and edges as relations. Those

factual triples are simple, and at the same time, they combined to a KG are a flexible data model with expressive power to describe complex circumstances and relationships.

Temporal Knowledge Graphs (TKG) have, in addition to the factual triple temporal relation-describing information, resulting in a quadruple. An example (Cristiano Ronaldo, plays for, Real Madrid, 2009-2018). TKG can also be referred to as dynamic, evolving, or time-varying graphs².

²All you need to know about temporal knowledge graphs - <https://analyticsindiamag.com/all-you-need-to-know-about-temporal-knowledge-graphs/>

Related Work

Research concerning the capability of NLP models acting as knowledge bases has already been conducted, as well as little research on the temporal capability of PLMs. The following is a summary of selected work progressing in similar directions to my research.

Petroni, Rocktäschel, Lewis, Bakhtin, Y. Wu, Miller, and Riedel (2019) argues that language models like BERT already have strong capabilities to retrieve factual knowledge without any fine-tuning, which means that these models already have high potential to eventually pass as unsupervised open-domain question answering systems. They base their claim on the fact that relational knowledge is already present in BERT without fine-tuning and can already compete with traditional NLP methods, although in some cases these models also had access to oracle knowledge. In addition, answering open domains performed very well compared to its selected baselines. And last but not least, that certain factual knowledge is learned much more easily through standard language model pre-training approaches than others.

However, Pörner, Waltinger, and Schütze (2019) has discovered a fact in Petroni, Rocktäschel, Lewis, Bakhtin, Y. Wu, Miller, and Riedel (2019)'s work that invalidates his claim that BERT already has a high potential to function as a knowledge base. Pörner, Waltinger, and Schütze (2019) argues that Petroni, Rocktäschel, Lewis, Bakhtin, Y. Wu, Miller, and Riedel (2019) has fallen into a fallacy because the datasets he used had many easily guessable entries. Examples of such easily guessable records would be '[Fiat Multipla] is produced by [Fiat]', '[Christmas Island] is named after [Christmas]' and '[Fulvio Tomizza] used to communicate in [Italian]'. By excluding just such records, a drastic decrease in the accuracy of BERT was observed (Pörner, Waltinger, and Schütze, 2019).

Roberts, Raffel, and Shazeer (2020) has researched how much knowledge is stored in the parameters through pre-training. T5 PLMs were fine-tuned to answer questions without access to external context or knowledge. Care was taken to fine-tune this without

inputting additional information or context in order to force the T5 model to answer questions based on 'knowledge' it had internalised during pre-training. It was concluded that performance scaled with model size and kept pace with open-domain systems that explicitly retrieved answers from an external knowledge source when answering questions. As a disclaimer it was also mentioned that only really big models with 11 billion parameters were competitive. In addition, Roberts, Raffel, and Shazeer (2020) has noted that knowledge is inexplicably distributed throughout the model, and if the model is uncertain about a question, realistic-looking answers are hallucinated. Within their trials the model was only tested based on "trivia"-style knowledge (Roberts, Raffel, and Shazeer, 2020).

Dhingra, Cole, Eisenschlos, Gillick, Eisenstein, and Cohen (2021) explored the question of whether LMs adequately learn the temporal scope of the encoded facts and how this knowledge can be better taught to LMs. These questions were explored based on three issues that Dhingra, Cole, Eisenschlos, Gillick, Eisenstein, and Cohen (2021) identified with temporal data. Averaging, which by generally ignorizing temporal metadata can lead to the model having low confidence in one of the correct answers by teaching the model temporal knowledge, which can lead to inconsistencies, e.g. "Lebron James plays for the Cavaliers / Lakers". Another issue is forgetting, which can be caused by the non-uniformity of the training data, creating a disparity in the timeliness of the documents. The number of new documents is much higher compared to older documents, because old documents can be updated and because over time more web documents are created than in the past. This leads to the model not remembering facts from underrepresentative time periods, making it worse at answering questions from the distant past. Finally, there is poor temporal calibration, which is caused by the obsolescence of the LM as the information with which the models are trained becomes less up-to-date, making it more likely that facts outside the training data frame will be queried. Dhingra, Cole, Eisenschlos, Gillick, Eisenstein, and Cohen (2021) has designed a method for mitigating recency that improves recall of seen facts from the training period. This is achieved by training the model with temporal context, which results in efficient refreshing that can be applied without starting from scratch (Dhingra, Cole, Eisenschlos, Gillick, Eisenstein, and Cohen, 2021).

To the best of my knowledge, no work has already researched the capability of PLMs of being temporal knowledge bases regarding temporal information of factual triples.

Method

In this chapter, the framework is presented in the scope of which the experiments were carried out. The different stages of the framework are then explained. The TKGQuestionMaster (QG) and its functionalities are described. The used text-to-text PLMs are presented as well as an overview of their special features. Last but not least, it is explained how the answers of the models were handled.

4.1 Framework

The experimental results were obtained following the B-procedure of the defined framework illustrated in Figure 4.1.

B-procedure involves three steps. The first step is based on TKGs' quadruples. TCBQ are generated by the TKGQuestionGenerator (QG) paired with their correct answers. In the second step, the questions are asked to selected text-to-text PLM, and their answers are stored. In the last step, their answers are compared with the correct answer, and the overall performance is evaluated.

The figure 4.1 also illustrates an A-procedure. In this procedure, the original motivation behind the QG is illustrated. It is to automatically generate TCBQ based on a KG in order to perform automated KG expansion to a TKG using a robust text-to-text PLM. However, this procedure is currently unrealistic due to the lack of PLM, which can work as a robust temporal knowledge base. Nevertheless, the procedure is included for completeness.

4.2 TKGQuestionGenerator

TKGQuestionGenerator (QG) is a pipeline with several predefined templates that can generate TCBQ based on KGs' triples. The templates are patterns that are used to

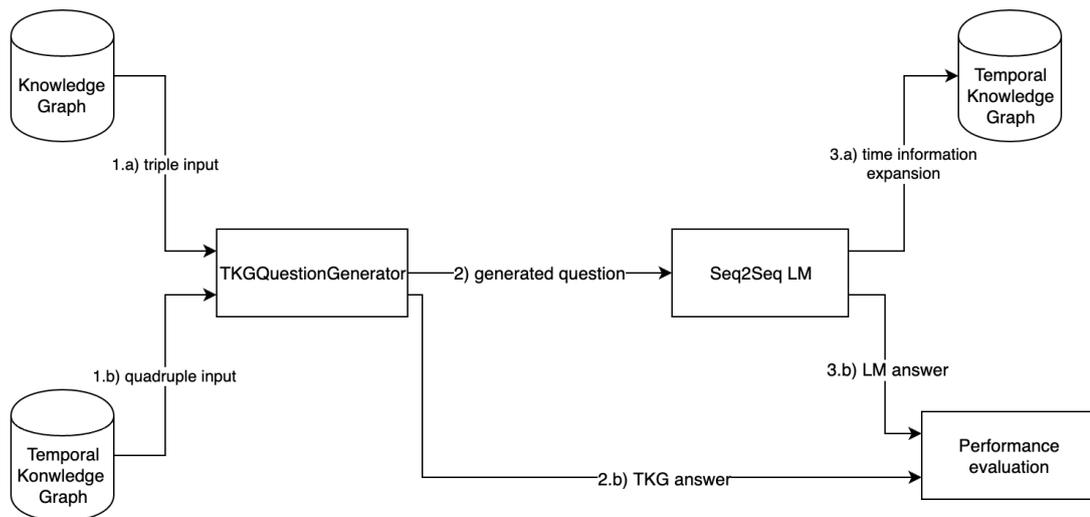


Figure 4.1: Experiment framework

guide the QG to generate TCBQ. QG was built upon the YAGO11k (Dasgupta, Ray, and P. Talukdar, 2018) data set. Therefore, it is well suited for it and for all KG triples where the predicates are similar to YAGO’s (F. Suchanek, Kasneci, and Weikum, 2007) predicates. Initially, the main goal is compatibility with as many different KGs as possible. Due to the lack of a definition for KGs (Ji, S. Pan, Cambria, Marttinen, and Yu, 2020) or standardisation of their triples’ formulation and the lack of resources, losses had to be accepted and QG’s compatibility is reduced to a limited number of KGs.

There are many TKG, for example, YAGO11k, WIKIDATA12k, icews05-15 and icews14 (Dasgupta, Ray, and P. Talukdar, 2018), and there are also already some TCBQ datasets, for instance, CronQUESTIONS (Saxena, Chakrabarti, and P. P. Talukdar, 2021). Nevertheless, there is, to the best of my knowledge, no library that generates TCBQ based on KG’s triples. Therefore the QG was created and can be found at the GitHub repository [hfock/TKGQuestionGenerator](https://github.com/hfock/TKGQuestionGenerator)¹.

QG incorporates templates with time indications listed in table 4.1 and templates without time indications listed in table 4.2. Time indications refer to the part in the question that tells the model that the question targets a year as an outcome, for instance, "In which year..." and "From which year until which year...".

Additionally, besides the swap-able time indication QG also incorporates two ways of predicate representation. It is possible to group together different forms of the same word by activating the in QG built-in mechanism of lemmatisation (Plisson, Lavrac, Mladenec, et al., 2004). By activating the lemmatisation, a predicate like 'plays for' is converted to 'play for'. This should allow triples such as (Cristiano Ronald, plays for, Real Madrid) to be generated into grammatically questions such as 'When did Cristiano

¹[hfock/TKGQuestionGenerator](https://github.com/hfock/TKGQuestionGenerator) - <https://github.com/hfock/TKGQuestionGenerator>

Question type	Closed Book Question Templates without time indication
Yes/No	Was/Did {Subject} {Predicate} {Object} in {Time}?
When	When was/did {Subject} {Predicate} {Object}?
Until	Until when was/did {Subject} {Predicate} {Object}?
Left Open	From when until {End Year} was/did {Subject} {Predicate} {Object}?
Right Open	From {Start Year} until when was/did {Subject} {Predicate} {Object}?
When to when	From when to when was/did {Subject} {Predicate} {Object}?
Duration	For how long was/did {Subject} {Predicate} {Object}?

Table 4.1: Closed Book Question Templates

Question type	Closed Book Question Templates with time indication
Yes/No	Was/Did {Subject} {Predicate} {Object} in the year {Year}?
When	In which year was/did {Subject} {Predicate} {Object}?
Until	Until which year was/did {Subject} {Predicate} {Object}?
Left Open	From which year until the year {End Year} was/did {Subject} {Predicate} {Object}?
Right Open	From the year {Start Year} until which year was/did {Subject} {Predicate} {Object}?
When to when	From which year until which year was/did {Subject} {Predicate} {Object}?
Duration	For how many years was/did {Subject} {Predicate} {Object}?

Table 4.2: Time indicated Closed Book Question Templates

Ronaldo play for Real Madrid?’ instead of ‘When did Cristiano Ronaldo plays for Real Madrid?’. Besides that QG trims ‘be’ or ‘have’ in front of a predicate. For instance, ‘is married to’ is converted to ‘married to’ or ‘has won prize’ is converted to ‘won prize’. Additionally, if a ‘be’ is trimmed ‘was’ is taken instead of the ‘did’ as the question word, and no lemmatisation is applied even if it is enabled. Otherwise, questions such as ‘When was Bill Gates mary to Melinda French Gates?’ happen. This is because without this rule, questions such as ‘When was Bill Gates mary to Melinda French Gates?’ might happen.

The most crucial information of a factual triple lies within the relation part when it comes to the automated creation of questions based on such triples. QG can only generate valid TCBQ based on the assumption that the relationship part of the triple is, for instance, an optional part like [be, have]) and a verb followed by a preposition such as ‘played for’ or ‘is married to’. Otherwise, the TCBQ might end up grammatically incorrect. For instance, given the triple (Yoshihiko Noda, Make optimistic comment, Japan) from the data set icews05-15 (Dasgupta, Ray, and P. Talukdar, 2018) would end up as ‘When did Yoshihiko Noda make optimistic comment Japan?’.

Additionally, it is possible to generate yes/no questions which should be answered by no. This feature was built in to evaluate if a PLM is certain about its knowledge because it should also be capable of realising if something is wrong. An easy way to evaluate this is by querying yes/no questions that are answered with a no, such as “Did Cristiano Ronaldo play for Real Madrid in 2022?”.

This paragraph is a disclaimer to use QG with caution. Some triples have ambiguous times. Ambiguous times happen to facts. For example, in the data set icews14 (Dasgupta, Ray,

and P. Talukdar, 2018) is a fact stating 'subject': 'South Korea', 'predicate': ['Criticize', 'denounce'], 'object': 'North Korea', 'time': '2014' which most likely happened also in other years than 2014. In such cases, the formulated question must be a yes/no question. Otherwise, it is not easy to check the correctness automatically.

4.3 Text-To-Text Pre-Trained Language Models

For the experiments and the performance evaluation two specific PLMs were chosen, T0_3B² and T0pp³.

For querying TCBQ, it is necessary to have a text-to-text framework. Therefore, only the T5 models (Raffel, Shazeer, Roberts, Katherine Lee, Narang, Matena, Y. Zhou, Li, and P. J. Liu, 2019) are capable, as well as all the models that are based on T5 models, such as the T0 models (Sanh, Webson, Raffel, Bach, Sutawika, Alyafeai, Chaffin, Stiegler, Scao, Raja, Dey, Bari, C. Xu, Thakker, S. Sharma, Szczechla, Kim, Chhablani, Nayak, Datta, J. Chang, Jiang, H. Wang, Manica, Shen, Yong, Pandey, Bawden, T. Wang, Neeraj, Rozen, A. Sharma, Santilli, Févry, Fries, Teehan, Biderman, L. Gao, Bers, Wolf, and Rush, 2021) from BigScience.

Based on the accessibility, usability, performance and limited resources, the two previously mentioned models of the T0 series were selected for the experiments. The easy accessibility was granted because of the service of Hugging Face⁴ who made this work possible in the first place. The usability was granted because the T0 series are already fine-tuned, so they are capable of answering TCBQ. For instance, the t5-3b-ssm model has the note "Note: This model should be fine-tuned on a question answering downstream task before it is usable for closed book question answering."⁵ The performance of the T0 series was advertised by Sanh, Webson, Raffel, Bach, Sutawika, Alyafeai, Chaffin, Stiegler, Scao, Raja, Dey, Bari, C. Xu, Thakker, S. Sharma, Szczechla, Kim, Chhablani, Nayak, Datta, J. Chang, Jiang, H. Wang, Manica, Shen, Yong, Pandey, Bawden, T. Wang, Neeraj, Rozen, A. Sharma, Santilli, Févry, Fries, Teehan, Biderman, L. Gao, Bers, Wolf, and Rush (2021)⁶ indicating that being 16x smaller, it still outperforms or matches GPT-3 (Brown, Mann, Ryder, Subbiah, Kaplan, Dhariwal, Neelakantan, Shyam, Sastry, Askell, Agarwal, Herbert-Voss, Krueger, Henighan, Child, Ramesh, Ziegler, J. Wu, Winter, Hesse, Mark Chen, Sigler, Litwin, Gray, Chess, J. Clark, Berner, McCandlish, Radford, Sutskever, and Amodei, 2020).

T0_3B was chosen because it is equally trained as T0⁷ but is based on a T5-LM XL model with 3 billion parameters while the other T0 models are based on a T5-LM XL model with 11 billion parameters.

²Hugging Face bigscience/T0_3B - https://huggingface.co/bigscience/T0_3B

³Hugging Face bigscience/T0pp - <https://huggingface.co/bigscience/T0pp>

⁴Hugging Face - <https://huggingface.co/>

⁵t5-3b-ssm - <https://huggingface.co/google/t5-3b-ssm>

⁶T0 BigScience Github Repository - <https://github.com/bigscience-workshop/t-zero>

⁷Hugging Face bigscience/T0 - <https://huggingface.co/bigscience/T0>

T0pp was chosen because BigScience advertises that the T0pp checkpoint has, on average, the best performance across on a variety of NLP tasks⁸.

T5

The T5 model created by Raffel, Shazeer, Roberts, Katherine Lee, Narang, Matena, Y. Zhou, Li, and P. J. Liu (2019) stands for "**T**ext-**T**o-**T**ext **T**ransfer **T**ransformer" and is the first transformer-based NLP model based on a text-to-text framework. The architecture is very similar to the original encoder-decoder-transformer implementation published by Vaswani, Shazeer, Parmar, Uszkoreit, Jones, Gomez, Kaiser, and Polosukhin (2017). The only differences are that the layer norm bias has been removed, the layer normalisation has been placed outside the residual path, and a different position embedding scheme is used (Raffel, Shazeer, Roberts, Katherine Lee, Narang, Matena, Y. Zhou, Li, and P. J. Liu, 2019).

The pre-training process is based on the Colossal Clean Crawled Corpus (C4) dataset produced by the authors as part of the T5 research. C4 is based on the Common Crawl Web Archive⁹, a massive collection of web-extracted text, with around 20 TB of text data added each month. The researchers took the massive collection of web-extracted text data with the actuality of April 2019. The massive corpora contain unusable text. Therefore, they removed inappropriate text passages using heuristics which are described in the C4 section of Raffel, Shazeer, Roberts, Katherine Lee, Narang, Matena, Y. Zhou, Li, and P. J. Liu (2019).

In creating the T5 model, the researchers focused primarily on measuring general language learning abilities by examining performance on many benchmarks, including machine translation, question answering, abstract summarisation and text classification.

In order for the researchers to use a single model to train all of these different tasks, all of them were converted into a text-to-text format. Initially, text prefixes were used for training to signal the different tasks. For example, "TL;DR:" (short for "too long, didn't read") to signal summary tasks.

Their generalised tasks were based on the "Natural Language Decathlon" (McCann, Keskar, Xiong, and Socher, 2018), which has a guideline that different NLP tasks have to be implemented in the same question-answer format under the condition that the models are doing multitasking by being able to handle all tasks simultaneously. The T5 researchers have taken liberties by fine-tuning their model separately for each task and supplementing the question format with task prefixes.

The objective of pre-training based on unlabelled data was defined as a 'denoising' objective as it achieves better performance rather than causal language modelling objective (Devlin, M. Chang, Kenton Lee, and Toutanova, 2019; Taylor, 1953). In a denoising goal, the model is trained to predict missing or otherwise corrupted tokens in the input. Here,

⁸Hugging Face T0 Model Card - <https://huggingface.co/bigscience/T0>

⁹Common Crawl - <https://commoncrawl.org/>

the researchers applied "masked language modelling" with the regularisation technique "word dropout" (Bowman, Vilnis, Vinyals, A. M. Dai, Jozefowicz, and S. Bengio, 2015) by randomly omitting 15% of the tokens in the input sequence.

T0

Sanh, Webson, Raffel, Bach, Sutawika, Alyafeai, Chaffin, Stiegler, Scao, Raja, Dey, Bari, C. Xu, Thakker, S. Sharma, Szczechla, Kim, Chhablani, Nayak, Datta, J. Chang, Jiang, H. Wang, Manica, Shen, Yong, Pandey, Bawden, T. Wang, Neeraj, Rozen, A. Sharma, Santilli, Févry, Fries, Teehan, Biderman, L. Gao, Bers, Wolf, and Rush (2021) created and published the T0 model series by exploring whether zero-shot generalisation can be induced by explicit multitask learning. This built on the publicly available T5 model by generating 100,000 additional steps with a standard language modelling goal. In addition, it was never trained to generate the input.

Like the T5 model, the T0 model has a text-to-text framework. After evaluating their experiments, they found strong zero-shot performance, with the T0 outperforming models up to $16\times$ larger in several standard data sets. (Sanh, Webson, Raffel, Bach, Sutawika, Alyafeai, Chaffin, Stiegler, Scao, Raja, Dey, Bari, C. Xu, Thakker, S. Sharma, Szczechla, Kim, Chhablani, Nayak, Datta, J. Chang, Jiang, H. Wang, Manica, Shen, Yong, Pandey, Bawden, T. Wang, Neeraj, Rozen, A. Sharma, Santilli, Févry, Fries, Teehan, Biderman, L. Gao, Bers, Wolf, and Rush, 2021)

The different extra data sets that were used to fine-tune the T5 base can be found in the training data section of the official Hugging Face model card¹⁰ of the T0 series.

4.4 Assessment of the Models' Answers

To ensure reproducibility, the Python functions used for the evaluation are listed in the 8 Appendix - Assessment of the Models' Answers Python Functions.

Answer Normalisation

By querying TCBQs to either T0pp or T0_3B their answer is always embedded in two extensions "<pad> " and "</s>", for instance, "<pad> 18 March 2012</s>". Before the response can be evaluated "<pad>" and "</s>" are trimmed, and all letters are lower cased such as 'Yes' becomes 'yes'. The used Python functions for this procedure are displayed in the listing 8.1 Normalisation of the Answer.

Valid Answer

During the evaluation, it is checked if the model's answer is valid and depending on the question type, the criteria vary if the answer is valid. In table 4.3 are all question types

¹⁰Hugging Face T0 Training Data Section - <https://huggingface.co/bigscience/T0pp#training-data>

listed with a formulation, the Python functions that were used and examples of what is seen as a valid answer.

Question type	Valid Answer	Listing of the used Python Function	Example
Yes/No	either "yes" or "no" without any extra text	8.2 Is Yes/No Validity Check	'yes', 'no'
When	a year with 3 or 4 digits standalone or embedded in a sentence the 4 digit year must start with 1, 2 or 3	8.3 Has Year Validity Check	'2012', '333', '1234', '.. in the year 2012 ..'
Until	a year with 3 or 4 digits standalone or embedded in a sentence the 4 digit year must start with 1, 2 or 3	8.3 Has Year Validity Check	'2012', '333', '1234', '.. in the year 2012 ..'
Left Open	a year with 3 or 4 digits standalone or embedded in a sentence the 4 digit year must start with 1, 2 or 3	8.3 Has Year Validity Check	'2012', '333', '1234', '.. in the year 2012 ..'
Right Open	a year with 3 or 4 digits standalone or embedded in a sentence the 4 digit year must start with 1, 2 or 3	8.3 Has Year Validity Check	'2012', '333', '1234', '.. in the year 2012 ..'
When to when	a least two years with 4 digits standalone or embedded in a sentence the 4 digit year must start with 1, 2 or 3	8.4 Has Two Years Validation and Extraction	'2000 - 2010', '.. was built 2000 and destroyed 2010.'
Duration	the 4 digit year must start with 1, 2 or 3 a numerical number or a formulation of such	8.5 Has Duration Validity Check	'10 years', '10', 'ten years', 'ten'

Table 4.3: Valid Answers: Used Functions and Examples for each Question Type

Correct Answer

When an answer is tagged as valid, the wanted information is extracted. The extraction differs from the question type. Yes/No questions do not need an extraction. For When, Until, Left Open and Right Open questions, the year within the answer is extracted. Regarding When to When questions, two years are extracted, and for the Duration questions, a number or a textual description of a number gets extracted. The textual description of a number is converted to a number with the Python library `word2number`¹¹

After all the comparable elements of the answers have been extracted, they are compared with the correct result depending on the Question Type. For Yes/No questions, the answer has to be precisely the correct "yes" or "no". For When questions, the answer has to include a year within the correct interval. The answer must be the same year for Until, Left Open and Right Open questions. The answer must include the exact start and the end year for When to When questions. For duration questions, the extracted period has to be the same number as the difference of the interval. For instance, given the quadruple <Bill Gates, is married to, Melinda Gates, 1994-2021> the correct answer would be the number 27, indicating 27 years.

An example for When questions, taking "When was Obama born?" and the answer of the model was "<pad> He was born in 2000</s>", acknowledging that a year with 3 or 4 digits is given then it is tagged as valid and only the year is extracted. In this case, it is "2000" while the rest of the answer is removed. After the wanted part of the answer is extracted, it is compared with the correct result, and in this case, the answer would be valid but incorrect because Obama was born in 1961.

¹¹Github - Word to Number by akshaynagpal - <https://github.com/akshaynagpal/w2n>

4. METHOD

In table 4.4 an overview is listed which describes with which function the desired part was extracted from the answer and with which function the result was compared with the correct answer.

Question type	Listing of the used Python Functions for Extraction	Listing of the used Python Functions for Correctness Check
Yes/No	-	8.8 Yes/No Correctness Check
When	8.7 Year Extraction	8.11 Is Year in Interval Correctness Check
Until	8.7 Year Extraction	8.9 Numbers Equal Correctness Check
Left Open	8.7 Year Extraction	8.9 Numbers Equal Correctness Check
Right Open	8.7 Year Extraction	8.9 Numbers Equal Correctness Check
When to when	8.4 Has Two Years Validation and Extraction	8.10 First and Last Years Equal Correctness Check
Duration	8.6 Duration Extraction	8.9 Numbers Equal Correctness Check

Table 4.4: Correct Answers: Used Extraction and Comparison Functions for each Question Type

Data analysis

In this chapter, the TKG YAGO11k is described, which was utilised for this work. The following will address the missing data. Finally, the questions generated by QG and based on YAGO11k are explained.

5.1 Chosen Temporal Knowledge Graph

The YAGO11k dataset from (Dasgupta, Ray, and P. Talukdar, 2018) was used for the empirical evaluation. YAGO (F. Suchanek, Kasneci, and Weikum, 2007) was automatically extracted from Wikipedia and unified with WordNet¹. The authors developed a combination of rule-based and heuristic methods to generate the knowledge base automatically. The empirical evaluation of the factual correctness indicated 95%. YAGO3 (Mahdisoltani, Biega, and F. M. Suchanek, 2015) adds 1 million new entities and 7 million new facts to the YAGO knowledge base by combining multilingual information with the English WordNet.

Dasgupta, Ray, and P. Talukdar (2018) extracted YAGO11k from the YAGO3 knowledge graph by selecting the ten most frequent temporal relationships. In creating YAGO11k, high connectivity was taken into account by recursively removing edges from entities that had only one mention in the subgraph. This fact played a role in the decision to use the dataset under the assumption that strongly connected entities enjoy higher representation, thereby assuming that the pre-training data for PLMs are likely to contain more information about just such strongly connected entities than about entities that have few connections in such graphs. In total, YAGO11k contains 20,509 triples and 10,623 entities. YAGO11k contains time annotations at the granularity of days, but for the experiments, the granularity was reduced to years.

¹WordNet - <https://wordnet.princeton.edu/>

5.2 Missing Data

Missing temporal-related data is part of the YAGO11k data set. The reason is unknown for the missing data. However, depending on the context, some imputation techniques are feasible.

5.2.1 Imputable Data

The subset of entries with the predicates <has won prize>, <created>, and <graduated from> often lacks the end year. The assumption is that winning a prize or graduating from university, or creating something is a point in time and not a period. However, there are some entries such as (Gerald M. Rubin, has won prize, Newcomb Cleveland Prize, 1983, 2000), (Rob Wagner, graduated from, University of Michigan, 1894, 1903) and (Barbara Adachi, created, San Francisco, 2003, 2008) indicating a period. Nevertheless, all the missing end years are imputed with the starting years for such entries. In total, 5245 entries are affected.

5.2.2 Not Imputable Data

The subset of entries with the predicates <is married to>, <works at>, <owns>, <plays for> and <affiliated to> often lacks the end year. Here is the assumption that the marriage, the employment relationship, the affiliation and the ownership are still ongoing and, therefore, the end year is indicated as #####.

For instance:

- (Athena Chu, is married to, Paul Wong (musician), 2012, #####)
- (Timothy Killeen (academic), works at, University of Illinois system, 2015, #####)
- (Yahoo!, owns, Polyvore, 2015, #####)
- (Miguel Lopes, plays for, Sporting Clube de Portugal, 2013, #####)
- (Richard Tsoi, is affiliated to, The Frontier (Hong Kong), 2008, #####)

A possible imputation would be the replacement of the gap filler '#####' with the year 2022, indicating the period is still ongoing and in at least until yet. This assumption might be correct for some entries but not for all, such as:

- (Edgar Speyer, is married to, Leonora Speyer, 1902, #####)
- (Berend George Escher, works at, University of Amsterdam, 1911, #####)
- (Frank McNeil, owns, Buffalo (NFL), 1920, #####)
- (Paulo Alves, plays for, Portugal national under-20 football team, 1989, #####)

Table 5.1: Summary - YAGO11k and Cleansed YAGO11k

	YAGO11k	Cleansed YAGO11k
Triples	20509	16734
Entities	10623	9838
Predicates	10	10

- (Roger B. Taney, is affiliated to, Federalist Party, 1828, ####)

Assuming that the death of a married couple is not the end of the marriage or the death of a person is not the end of their affiliation then it would be legit to impute the current year. Nevertheless, the imputation technique is not arguable for entries indicating ownership or the employment relationship and the apparent impossible relation of a football player playing for the under-20 team for over 30 years. Due to these reasons, all related entries are dropped from the data. In total, 3752 entries are affected.

Besides not having the end year there are also several entries without concrete dates. These entries rather have an approximations when the specific event happened. For instance, (Yue Fei, created, Xing Yi Quan, 12##, 19##). In these specific cases the data was also dropped and in total, 23 entries were affected.

Table 5.2: Predicate Summary - YAGO11k and Cleansed YAGO11k

predicate	YAGO11k	Cleansed YAGO11k
was born in	3341	3340
plays for	4787	3312
has won prize	3307	3307
created	1943	1921
died in	1643	1643
is married to	2312	1377
is affiliated to	1388	631
graduated from	630	630
owns	750	364
works at	408	209

5.3 Generated Questions

To gain a more enlightened insight into PLM several experiments have been conducted which are based on TCBQ more details about the experiments' results can be found in section 6 Empirical Evaluation. One field of interest is the formulation of TCBQ and how much this influences the performance. This was researched by creating four different versions of question sets abbreviated as V1, V2, V3 and V4. The main difference of the versions are if the predicate was subjected to lemmatisation or the question includes a

time indication. In the table 5.3 is a listing of the version specific modifications and for each version a when-question example based on the triple (Cristiano Ronaldo, plays for, Real Madrid).

Version	Lemmatisation	Time Indication	Example
V1	False	False	When did Cristiano Ronaldo plays for Real Madrid?
V2	True	False	When did Cristiano Ronaldo play for Real Madrid?
V3	False	False	In which year did Cristiano Ronaldo plays for Real Madrid?
V4	True	True	In which year did Cristiano Ronaldo play for Real Madrid?

Table 5.3: Question Versions

Another area of interest is how robust the PLMs are with their knowledge. The experiment is based on yes/no questions to investigate this particular question. For this purpose, questions have been created that query the PLM yes/no questions and where the questions that should be answered with 'no' are very close to the correct years. Two robust yes/no question sets were created for each question version. The first question set contains yes/no questions for each possible correct year if the input has an interval; otherwise, there is only one question that can be answered with a yes. For example, the quadruple (Bill Gates, is married to, Melinda Gates, 1994-2021) results in "Was Bill Gates married to Melinda Gates in 1994?", "Was Bill Gates married to Melinda Gates in 1995?", ..., "Was Bill Gates married to Melinda Gates in 2021?". The second set of questions contains only the first year of the correct years, even if an interval is given. In addition to the yes/no questions that are answered with a yes, four yes/no questions are added that should be answered with a no, two consecutive years before the first year and two consecutive after the last year. For example, "Was Bill Gates married to Melinda Gates in 1993?", "Was Bill Gates married to Melinda Gates in 1992?", "Was Bill Gates married to Melinda Gates in 2022?" and "Was Bill Gates married to Melinda Gates in 2023?".

12 Question sets were generated by the QG, and they are all listed with their abbreviations which will be used as references at the table 5.4. The indication 'ALL' in the column Question describes that all possible question types are incorporated, whereby the indication 'YES/NO_ROBUST' describes that these question sets do only consist of Yes/No questions but in contrast to the 'ALL' questions additionally falsely Yes/No questions are added.

	Question Abbreviation	Question	Lemmatisation	Time indication
1	QV1	YAGO11k_ALL_V1	False	False
2	QV2	YAGO11k_ALL_V2	True	False
3	QV3	YAGO11k_ALL_V3	False	True
4	QV4	YAGO11k_ALL_V4	True	True
5	QV1R	YAGO11k_YES/NO_ROBUST_V1	False	False
6	QV1RCI	YAGO11k_YES/NO_ROBUST_V1_COMPLETE_INTERVAL	False	False
7	QV2R	YAGO11k_YES/NO_ROBUST_V2	True	False
8	QV2RCI	YAGO11k_YES/NO_ROBUST_V2_COMPLETE_INTERVAL	True	False
9	QV3R	YAGO11k_YES/NO_ROBUST_V3	False	True
10	QV3RCI	YAGO11k_YES/NO_ROBUST_V3_COMPLETE_INTERVAL	False	True
11	QV4R	YAGO11k_YES/NO_ROBUST_V4	True	True
12	QV4RCI	YAGO11k_YES/NO_ROBUST_V4_COMPLETE_INTERVAL	True	True

Table 5.4: Question listing and Abbreviations

All generated questions are based on the cleansed YAGO11k data set, the given question formulation options listed at table 4.1 and 4.2 by QG and the different versions listed at table 5.3. Additionally all questions sets can be found in the repository [hfock/TKGQuestionGenerator](https://github.com/hfock/TKGQuestionGenerator)².

²[hfock/TKGQuestionGenerator](https://github.com/hfock/TKGQuestionGenerator) - <https://github.com/hfock/TKGQuestionGenerator>

Empirical Evaluation

In this chapter, all survey results are listed, and conspicuous facts are described. Limitations are attached at the end, explaining why both models were not surveyed to the same extent.

6.1 Evaluation

General notes regarding tables 6.3, 6.4, 6.5 and 6.6 they miss the absolute numbers of correctly and validly answered questions because of clarity reasons. Therefore, a summary table for each question version has been included in the appendix. The summary tables are 1, 2, 3, 4 and 5. For all mentioned tables an empty cell indicates that the predicate was excluded for the certain question type. Each percentage in the tables reflecting the results was calculated by dividing the number of valid or correct answers by the total of the corresponding question type or predicate, rounding to the fourth decimal place, and multiplying by 100.

The QV3 question set was taken on purpose for the only big model evaluation because it performed on average the best.

All the results can be found in the repository [hfock/TKGQuestionGenerator](https://github.com/hfock/TKGQuestionGenerator)¹.

¹[hfock/TKGQuestionGenerator](https://github.com/hfock/TKGQuestionGenerator) - <https://github.com/hfock/TKGQuestionGenerator>

6.1.1 T0_3B Regarding QV1 - 4

Table 6.1 shows all the results of the survey of the T0_3B model regarding the question sets QV1 - 4. It can be seen that the addition of the time indicator for Yes/No questions triggered a sharp increase in the correctness of about 20%. For each question type that only asked about one year, the addition of the time indicator caused an increase in performance. Whereby, the questions that targeted a period of time experienced a decrease in performance due to the time indicator. Regarding the lemmatisation of the predicates, there is no clear trend. While non-lemmatisation increased performance for the Yes/No question by approximately 7%, it is not clear for the other questions rather it is beneficial or not. For instance, the When questions saw an increase for non-lemmatisation without the time indicator whereby the lemmatisation with the time indicator saw an increase. However, the difference is negligible. A similar behaviour could also be observed for the Duration questions and the Until questions.

Question Type	Size	Correct Answer	Correct Answer %	Valid Answer	Valid Answer %
Yes/No V1	16734	4536	27.11	15373	91.86
Yes/No V2	16734	3536	21.13	14993	89.6
Yes/No V3	16734	8097	48.39	15228	91.0
Yes/No V4	16734	6757	40.38	15021	89.75
When V1	16734	1285	7.68	16656	99.53
When V2	16734	1279	7.64	16647	99.48
When V3	16734	1437	8.59	16731	99.98
When V4	16734	1446	8.64	16731	99.98
Until V1	6492	232	3.57	6329	97.49
Until V2	6492	235	3.62	6273	96.63
Until V3	6492	264	4.07	6489	99.95
Until V4	6492	253	3.9	6489	99.95
Left Open V1	6492	124	1.91	6463	99.55
Left Open V2	6492	124	1.91	6451	99.37
Left Open V3	6492	256	3.94	6492	100.0
Left Open V4	6492	272	4.19	6492	100.0
Right Open V1	6492	215	3.30	4233	65.2
Right Open V2	6492	246	3.79	4622	71.2
Right Open V3	6492	422	6.5	6492	100.0
Right Open V4	6492	439	6.76	6492	100.0
When to When V1	6492	6	0.09	920	14.17
When to When V2	6492	8	0.12	1233	18.99
When to When V3	6492	0	0.0	0	0.0
When to When V4	6492	0	0.0	0	0.0
Duration V1	6492	307	4.73	5189	79.93
Duration V2	6492	223	3.42	5053	77.83
Duration V3	6492	140	2.16	6451	99.37
Duration V4	6492	141	2.17	6442	99.22

Table 6.1: Evaluation of T0_3B on QV1-4

Whereas the time indicator which is explained in section 5.3 Generated Questions decreased the overall performance of the duration questions and at the same time they increased the validity of the responses to nearly 100% from about 80%. Another conspicuous feature is the general high validity rate of the model with regard to the question set, as it is almost always 100% with the exception of V1 and V2 of Right Open

questions, which only reached around 80% and the When to When questions reached with V3 and V4 0% and with V1 and V2 20%.

6.1.2 T0pp vs T0_3B Regarding QV3

Table 6.2 shows the interview results of both models T0_3B and T0pp with respect to question set QV3. Based on the validity rate, it can be seen that the larger model has a generally better understanding of how to respond to those questions even the When to When questions resulted in a 64.4% validity rate. While the small model had no understanding at all regarding the When to When questions that was expressed by a percentage of 0.

For the Yes/No questions, there is an increase in the validity rate of about 8%, but there is also a decrease in the correctness rate of about 10%. While the Yes/No and Until correctness rate decreased, the correctness rate for all other question types increased steadily.

Model	Question Type	Size	Correct Answer	Correct Answer %	Valid Answer	Valid Answer %
T0pp	Yes/No V3	16734	6515	38.93	16545	98.87
	When V3	16734	1590	9.5	16728	99.96
	Until V3	6492	239	3.67	6492	100.0
	Left open V3	6492	490	7.55	6492	100.0
	Right open V3	6492	423	6.52	6492	100.0
	When to When V3	6492	39	0.6	4181	64.4
	Duration V3	6492	462	7.12	6462	99.53
	T0_3B	Yes/No V3	16734	8097	48.39	15228
When V3		16734	1437	8.59	16731	99.98
Until V3		6492	264	4.07	6489	99.95
Left Open V3		6492	256	3.94	6492	100.0
Right Open V3		6492	422	6.5	6492	100.0
When to When V3		6492	0	0.0	0	0.0
Duration V3		6492	140	2.16	6451	99.37

Table 6.2: Comparison of T0pp to T0_3B on QV3

6.1.3 Grouped by Predicate T0_3B

Grouping predicates gives insight into whether the model understands different relationships better or worse. The following analysis refers to tables 6.3 and 6.4.

There is a general trend in the Yes/No questions that the relationship 'was born in' achieved the lowest validity rate. In contrast to the other predicates, which score over 90%, the validity rate for 'was born in' in V1 and V2 is 79% and for V3 and V4 70%.

The When, Until, Left Open questions were generally understood almost perfectly, and all percentages stay in the neighbourhood of 100. However, the T0_3B did not have a relatively high validity rate with the V1 and V2 for Right Open questions what is not due to all predicates. For example, the predicate "is affiliated to" only reached 50% Valid Answer. Whereas for the predicate "graduated from", 100% Valid Answer were given.

Although T0_3B did not achieve a perfect validity rate with the Right Open V1 and V2 questions, 100% was achieved with the Right Open questions for V3 and V4. The time indicator improved the validity rate in general except for the When to When questions. Here, the time indicator must have served as a confounding component because a validity rate of 0% was achieved for each predicate. However, without the time indicator, T0_3B was able to answer at least some When to When questions regarding specific predicates validly. For example, 'is affiliated to' was answered for V1 and V2 with a 59% validity rate, whereas 'graduated from' was answered with a 0% validity rate.

The T0_3B can answer certain predicates better than others. For example, the predicate that performed best in the Yes/No questions was 'has won prize' with about 50% for V1 and V2 and with about 75% for V3 and V4, whereas the predicate 'is affiliated to' could only be answered correctly with 1%. However, it should be noted that in all versions of the When questions 'affiliated by' was answered correctly with about 12-15% and 'has won prize' dropped to the level of about 2%.

Predicate Version	Yes/No (%)	When (%)	Until (%)	Left Open (%)	Right Open (%)	When to When (%)	Duration (%)
created V1	95.78	99.11	98.6	100.0	91.14	0.47	43.36
created V2	93.02	99.11	97.67	100.0	93.01	1.17	36.83
created V3	96.41	99.83	99.3	100.0	100.0	0.0	97.67
created V4	95.67	99.83	99.3	100.0	100.0	0.0	95.8
died in V1	99.27	100.0					
died in V2	98.42	100.0					
died in V3	99.45	100.0					
died in V4	99.09	100.0					
graduated from V1	98.89	100.0	100.0	100.0	100.0	0.0	100.0
graduated from V2	96.98	100.0	100.0	100.0	100.0	0.0	97.85
graduated from V3	94.28	100.0	100.0	100.0	100.0	0.0	100.0
graduated from V4	92.22	100.0	100.0	100.0	100.0	0.0	100.0
has won prize V1	98.19	99.97	100.0	100.0	86.27	0.98	89.22
has won prize V2	92.92	99.94	100.0	100.0	98.04	0.98	75.49
has won prize V3	99.36	100.0	100.0	100.0	100.0	0.0	100.0
has won prize V4	97.91	100.0	100.0	100.0	100.0	0.0	100.0
is affiliated to V1	98.26	99.37	94.58	97.13	49.6	59.48	65.23
is affiliated to V2	98.26	99.37	94.58	97.13	49.6	59.48	65.23
is affiliated to V3	98.72	100.0	100.0	100.0	100.0	0.0	99.83
is affiliated to V4	98.72	100.0	100.0	100.0	100.0	0.0	99.83
is married to V1	84.24	99.06	99.19	99.33	77.62	1.09	76.74
is married to V2	84.24	99.06	99.19	99.33	77.62	1.09	76.74
is married to V3	90.27	100.0	100.0	100.0	100.0	0.0	99.71
is married to V4	90.27	100.0	100.0	100.0	100.0	0.0	99.71
owns V1	90.38	100.0	95.28	100.0	72.58	17.73	57.62
owns V2	89.84	100.0	93.91	100.0	78.95	24.65	59.0
owns V3	93.96	100.0	100.0	100.0	100.0	0.0	97.78
owns V4	92.86	100.0	100.0	100.0	100.0	0.0	97.50
plays for V1	92.78	98.7	97.13	99.94	57.22	11.31	89.69
plays for V2	89.01	98.46	95.71	99.61	67.11	16.66	87.15
plays for V3	94.02	100.0	100.0	100.0	100.0	0.0	99.46
plays for V4	90.34	100.0	100.0	100.0	100.0	0.0	99.46
was born in V1	79.07	100.0					
was born in V2	79.07	100.0					
was born in V3	69.78	100.0					
was born in V4	69.78	100.0					
works at V1	100.0	100.0	100.0	100.0	65.55	43.54	90.42
works at V2	100.0	100.0	100.0	99.52	74.64	95.22	84.21
works at V3	100.0	100.0	100.0	100.0	100.0	0.0	100.0
works at V4	100.0	100.0	100.0	100.0	100.0	0.0	100.0

Table 6.3: Valid Answer percentage - QV1-4 Grouped by Predicate - T0_3B

6. EMPIRICAL EVALUATION

Predicate Version	Yes/No (%)	When (%)	Until (%)	Left Open (%)	Right Open (%)	When to When (%)	Duration (%)
created V1	33.16	9.53	3.50	2.33	3.73	0.0	1.40
created V2	18.69	9.99	2.80	2.33	3.96	0.0	1.17
created V3	47.83	10.31	3.26	2.80	4.2	0.0	2.80
created V4	32.33	10.26	3.03	2.56	4.2	0.0	2.33
died in V1	9.31	1.89					
died in V2	2.13	2.25					
died in V3	47.34	1.22					
died in V4	24.27	1.16					
graduated from V1	26.83	1.75	0.0	2.15	2.15	0.0	2.15
graduated from V2	14.76	1.11	0.0	3.23	2.15	0.0	3.23
graduated from V3	53.97	1.9	1.08	3.23	1.08	0.0	10.75
graduated from V4	31.59	2.06	1.08	3.23	1.08	0.0	15.04
has won prize V1	56.34	2.18	4.9	0.0	4.9	0.0	4.9
has won prize V2	51.1	2.23	5.88	2.94	4.9	0.0	2.94
has won prize V3	78.86	2.39	0.98	4.9	5.88	0.0	7.84
has won prize V4	71.73	2.48	0.98	5.88	6.85	0.0	7.84
is affiliated to V1	0.79	11.57	1.44	3.03	0.0	0.0	0.48
is affiliated to V2	0.79	11.57	1.44	3.03	0.0	0.0	0.48
is affiliated to V3	1.27	14.74	1.59	3.03	2.39	0.0	1.11
is affiliated to V4	1.27	14.74	1.59	3.03	2.39	0.0	1.11
is married to V1	12.49	14.02	1.69	1.03	1.03	0.0	1.60
is married to V2	12.49	14.02	1.69	1.03	1.03	0.0	1.60
is married to V3	34.57	20.91	1.76	0.88	2.35	0.0	4.26
is married to V4	34.57	20.91	1.76	0.88	2.35	0.0	4.26
owns V1	5.77	21.98	0.83	0.0	1.66	0.0	0.27
owns V2	3.57	21.43	1.39	0.0	1.66	0.0	0.27
owns V3	13.74	21.15	2.22	0.54	3.05	0.0	1.11
owns V4	9.07	21.15	1.66	0.83	3.05	0.0	1.11
plays for V1	24.82	16.0	5.2	2.36	5.17	0.18	8.01
plays for V2	14.49	15.4	5.35	2.23	6.11	0.24	5.56
plays for V3	38.83	17.41	6.08	6.05	10.13	0.0	1.15
plays for V4	31.34	17.59	5.83	6.52	10.63	0.0	1.15
was born in V1	20.48	2.6					
was born in V2	20.48	2.6					
was born in V3	47.75	1.85					
was born in V4	47.75	1.85					
works at V1	4.78	11.95	2.39	0.48	0.48	0.0	1.44
works at V2	2.39	13.4	1.44	0.48	0.0	0.0	0.96
works at V3	17.7	14.82	2.39	1.44	1.91	0.0	1.44
works at V4	7.66	15.31	2.39	0.96	1.44	0.0	0.96

Table 6.4: Correct Answer percentage - QV1-4 Grouped by Predicate - T0_3B

6.1.4 Grouped by Predicate T0pp vs T0_3B

It can be seen that in each case the larger model answered the questions with a higher validity rate. Even for the When to When questions, a 100% validity rate was achieved in some cases. Whereby 'plays for' was most misleading for T0pp, as only a validity rate of 40% was achieved.

Nevertheless, the larger model has almost the same performance as the small model when it comes to the correctness rate regarding the When to When questions namely almost all were answered incorrectly.

For the larger model, the same situation can be observed as for the small model. The predicate 'has won prize' also performs very well for the Yes/No questions while for the When questions it performs very poorly. However, the inverted circumstance with the predicate 'is affiliated by' is not observed.

Model	Predicate Version	Yes/No (%)	When (%)	Until (%)	Left Open (%)	Right Open (%)	When to When (%)	Duration (%)
T0pp	created V3	99.64	99.95	100.0	100.0	100.0	96.97	99.77
	died in V3	99.7	99.7					
	graduated from V3	100.0	100.0	100.0	100.0	100.0	100.0	96.77
	has won prize V3	100.0	100.0	100.0	100.0	100.0	100.0	100.0
	is affiliated to V3	99.83	100.0	100.0	100.0	100.0	94.42	99.83
	is married to V3	100.0	100.0	100.0	100.0	100.0	90.9	100.0
	owns V3	94.78	100.0	100.0	100.0	100.0	62.05	93.63
	plays for V3	99.49	100.0	100.0	100.0	100.0	39.66	99.97
	was born in V3	95.81	100.0					
	works at V3	100.0	100.0	100.0	100.0	100.0	97.13	99.52
T0_3B	created V3	96.41	99.83	99.3	100.0	100.0	0.0	97.67
	died in V3	99.45	100.0					
	graduated from V3	94.28	100.0	100.0	100.0	100.0	0.0	100.0
	has won prize V3	99.36	100.0	100.0	100.0	100.0	0.0	100.0
	is affiliated to V3	98.72	100.0	100.0	100.0	100.0	0.0	99.83
	is married to V3	90.27	100.0	100.0	100.0	100.0	0.0	99.71
	owns V3	93.96	100.0	100.0	100.0	100.0	0.0	97.78
	plays for V3	94.02	100.0	100.0	100.0	100.0	0.0	99.46
	was born in V3	69.78	100.0					
	works at V3	100.0	100.0	100.0	100.0	100.0	0.0	100.0

Table 6.5: Valid Answer percentage - QV3 Grouped by Predicate - T0pp vs T3_3B

Model	Predicate Version	Yes/No (%)	When (%)	Until (%)	Left Open (%)	Right Open (%)	When to When (%)	Duration (%)
T0pp	created V3	34.98	10.41	2.1	6.52	9.09	0.47	2.1
	died in V3	92.45	2.5					
	graduated from V3	31.75	1.9	1.08	3.23	6.45	0.0	1.08
	has won prize V3	54.25	2.87	2.94	3.92	12.75	0.0	4.9
	is affiliated to V3	19.97	15.37	2.71	7.81	5.42	0.32	2.07
	is married to V3	17.43	27.16	2.86	3.74	4.18	0.22	3.01
	owns V3	40.38	22.53	3.32	3.05	4.71	0.83	1.11
	plays for V3	29.70	17.21	4.59	10.10	7.59	0.88	11.7
	was born in V3	22.99	2.65					
	works at V3	31.1	14.35	2.87	4.78	2.87	0.0	0.96
T0_3B	created V3	47.83	10.31	3.26	2.8	4.2	0.0	2.8
	died in V3	47.34	1.22					
	graduated from V3	53.97	1.9	1.08	3.23	1.08	0.0	10.75
	has won prize V3	78.86	2.39	0.98	4.9	5.88	0.0	7.84
	is affiliated to V3	1.27	14.74	1.59	3.03	2.39	0.0	1.11
	is married to V3	34.57	20.91	1.76	0.88	2.35	0.0	4.26
	owns V3	13.74	21.15	2.22	0.54	3.05	0.0	1.11
	plays for V3	38.83	17.41	6.08	6.05	10.13	0.0	1.15
	was born in V3	47.75	1.85					
	works at V3	17.7	14.82	2.39	1.44	1.91	0.0	1.44

Table 6.6: Correct percentage - QV3 Grouped by Predicate - T0pp vs T3_3B

6.1.5 Yes/No Robust - T0_3B

In this section, the analysis refers to table 6.7 in comparison to a specific section from table 6.1. Table 6.7 displays the results regarding the Yes/No Robust questions and table 6.1 displays the evaluation of the T0_3B regarding all question types but within this analysis only the Yes/No question type section is considered.

In order to address the two Yes/No Robust variants more efficiently, the question sets that refer to the complete interval are abbreviated as QV[1-4]RCI, and question sets that only consider the first year of the interval are abbreviated as QV[1-4]R in the following as it is stated in the table 5.4 Question listing and Abbreviations.

The QV[1-4]RCI generally performs worse than QV[1-4]R. On average, the performance loses 20%. When comparing the Yes/No questions in relation to QV1, QV2 and QV1R, QV2R, a substantial increase in Correct Answer can be observed. On average, the performance was increased by 34%. In contrast, the versions equipped with the time indicator perform worse in the case of QV3 vs QV3R with about - 5% and in the case of QV4 vs QV4R only slightly better with about 7%.

The column Correct Entity (%) in table 6.7 indicates the percentage of entities for which all related questions have been answered correctly. It can be seen that T0_3B did not achieve a performance of only one per cent for any of the versions, while some of the percentages of generally correctly answered questions exceeded 50%.

	Valid Answer	Correct Answer	Correct Answer %	Entity Count	Correct Entity	Correct Entity %
Yes/No Robust Complete Interval V1	200571	62805	31.31	16734	101	0.6
Yes/No Robust V1	76900	48042	57.42	16734	125	0.75
Yes/No Robust Complete Interval V2	181421	59526	29.68	16734	69	0.41
Yes/No Robust V2	75034	49474	59.13	16734	96	0.57
Yes/No Robust Complete Interval V3	188307	63962	31.89	16734	62	0.37
Yes/No Robust V3	76204	36624	43.76	16734	95	0.57
Yes/No Robust Complete Interval V4	186232	59304	29.57	16734	82	0.49
Yes/No Robust V4	75168	39903	47.69	16734	121	0.72

Table 6.7: Summary Yes/No Robust - T0_3B

6.2 Limitations

As already mentioned in section 4.3 Text-To-Text Pre-Trained Language Models the available resources for this thesis are limited and can be separated into two specific limitations namely hardware and time.

Hardware limitations refer to the available computing power. In the context of this work, access existed to servers provided by a Google Colab Pro+² subscription. However, these servers also have limits, and the required text-to-text PLM are disproportionately huge. This circumstance set the seals on that only one model with 11 billion parameters the T0pp has been used for the experiments and for a selective choice of questions.

In the table 6.8 are the servers' hardware specifications listed. The first two rows display the specifications of the servers used for evaluating the T0_3B model, and the third row

²Google Colab

Google Colab Runtime	CPU	RAM	GPU	GPU DRIVER
First possible Runtime for T0_3B Hardware Accelerator: GPU Runtime shape: High RAM Background execution: enabled	Model name: Intel(R) Xeon(R) CPU MHz: 2299.998 CPU(s): 8 Architecture: x86_64	54.8 GB	Tesla P100-PCIE Memory: 16280MiB	NVIDIA-SMI 460.32.03 Driver Version: 460.32.03 CUDA Version: 11.2
Second possible Runtime for T0_3B Hardware Accelerator: GPU Runtime shape: High RAM Background execution: enabled	Model name: Intel(R) Xeon(R) CPU MHz: 2000.178 CPU(s): 8 Architecture: x86_64	54.8 GB	Tesla V100-SXM2 Memory: 16160MiB	NVIDIA-SMI 460.32.03 Driver Version: 460.32.03 CUDA Version: 11.2
Runtime for T0pp Hardware Accelerator: None Runtime shape: High RAM Background execution: enabled	Model name: Intel(R) Xeon(R) CPU MHz: 2199.998 CPU(s): 8 Architecture: x86_64	54.8 GB	-	-

Table 6.8: Google Colab Pro+ Hardware Specifications

displays the specifications of the server used for evaluating the T0pp model. T0pp has a size of 41.5GB. Thus, it was too big to compute on the GPUs Google Colab Pro+ could provide. Even though the CPU MHz are higher for the first GPU hardware accelerated server, it was not possible to execute a big model on GPU hardware accelerated servers because Google Colab terminates runtimes if it occupy hardware which is not in use.

Time limitations refers to the start time and the deadline of this thesis which is in total a span of two months. In table 6.9, the processing time for every question set is stated. The big difference in time variations might be caused by different GPUs Google Colab Pro+ provided during the tests. Unfortunately, the machine specs were not always noted. Therefore, it is not possible to derive any conclusions based on these times. Because of the the limited time and the 72 hours it took the T0 model to process the QV3 dataset, the large model was only interviewed once.

Question Set	Model	Size	Processing Time
QV1	T0_3B	65928	2 h 10 min 4 sec
QV2	T0_3B	65928	2 h 13 min 23 sec
QV3	T0_3B	65928	1 h 28 min 58 sec
	T0pp		71 h 39 min 32 sec
QV4	T0_3B	65928	1 h 31 min 7 sec
QV1R	T0_3B	83670	2 h 46 min 22 sec
QV1RCI	T0_3B	200571	5 h 11 min 24 sec
QV2R	T0_3B	83670	2 h 7 min 10 sec
QV2RCI	T0_3B	200571	5 h 15 min 33 sec
QV3R	T0_3B	83670	2 h 48 min 58 sec
QV3RCI	T0_3B	200571	4 h 44 min 12 sec
QV4R	T0_3B	83670	1 h 55 min 54 sec
QV4RCI	T0_3B	200571	6 h 53 min 3 sec

Table 6.9: Processing Time per Question Set

Discussion, Ethics and Future Work

In the penultimate chapter, the careful consideration of the conclusion and an anomaly that occurred during the evaluation are discussed first. This is followed by a reflection on the ethical component that needs to be taken into account if PLMs are to be used as knowledge bases in everyday life. Finally, an outlook on future work is given.

7.1 Discussion

Reservations to the conclusion must be acknowledged. Due to the limited computing power and time, the two models were not investigated to the same extent. The larger model could only be examined based on one question set.

In addition, some improvements could be made to increase the models' performance, but they could not be implemented in this work. Fine-tuning for each question type is one possibility. Another possibility would be to additionally pre-train the models using data enriched with temporal context as Dhingra, Cole, Eisenschlos, Gillick, Eisenstein, and Cohen (2021) suggested. Furthermore, without considering resources, the evaluation of a much larger model would be possible. While this work was being processed, a model 58 times larger than the T0_3B¹ model was trained by the same creators of the T0 series.

It was often shown that training a more extensive model on a larger data set results in better performance (Radford, J. Wu, Child, Luan, Amodei, Sutskever, et al., 2019; Hestness, Narang, Ardalani, Damos, Jun, Kianinejad, Patwary, Y. Yang, and Y. Zhou, 2017). Surprisingly, the smaller model performs better in the Yes/No questions, but

¹Supercomputer to train 176-billion-parameter open-source AI language model - https://www.theregister.com/2022/03/25/supercomputer_language_ai/

there is a circumstance that could explain this. Interestingly, the smaller model almost always achieved a higher correctness rate for the predicate 'has won prize', which can be derived from the table 6.6. In this case, the following hypothesis is that the smaller model partly neglected the temporal component and only answered whether a particular person won this prize. Since the answer to all these questions is always yes, answering the more straightforward question without the temporal component, so to speak, whether the person won this prize at all, can lead to the same correct answer resting on a false foundation. This claim also seems to be reinforced by the fact that the overall validity rate of the smaller model is lower for Yes/No questions relative to the validity rate of the bigger model, which can be derived from the table 6.5. Therefore, it can be assumed that a larger model also leads to better results in this case, even though a question type from the larger model performed worse.

7.2 Ethical Concerns

From my perspective, PLM will eventually gain the ability to act as robust temporal and factual knowledge bases as long as Moore's law² continues to hold, and humanity's thirst for knowledge does not dry up.

One concern must be considered and resolved until PLMs are eventually used as super mature Siri and Cortana-like assistants. The knowledge will be given to these PLMs based on the pre-training phase by feeding them with all the knowledge of the world. However, the knowledge of the world is not summarised and validated by a board but drawn voraciously from the vast expanses of the Internet, just like Common Crawl does. The Internet is like an organism constantly growing, and each interacting individual serves as fodder. As from the saying, you are what you eat, it behaves similarly to the PLMs. Humankind is progressive, ingenious and tolerant and at the same time regressive, profane and intolerant, generating very valuable and very borderline content as well as everything in between and beyond these borders. Researchers already point out that past problematic views of humanity that are well and strongly countered today, such as racism and sexism, have been absorbed by the PLM. The T0 series that was surveyed in this work also have bias and fairness issues. For instance, "Is the earth flat?" is answered with "yes", as well as "Do vaccines cause autism?"³.

Under the assumptions that questions of PLMs are always answered with only one answer instead of answer options and the sources on which the answer is based are not made recognisable, the knowledge of PLM reflects the whole unfiltered internet and that PLMs assistants are used similar or even more like the Google search engine today, and its answers are generally accepted and considered valid, then there is a possibility that false

²Moore's Law is the observation that the number of transistors on integrated circuits doubles approximately every two years which is equivalent to an exponential growth of computing power.

³Bias and Fairness T0 models - <https://huggingface.co/bigscience/T0#bias-and-fairness>

truths or false views are propagated. Source research, which is possible when Googling, would not be possible.

These circumstances could give rise to communities of believers who represent false truths or false views and can theoretically invoke the fact that the cumulative knowledge of humanity expressed by the PLMs reinforces their views.

In short, more research needs to be done regarding preparing and monitoring the pre-training content before the quality of PLMs becomes an integral part of everyday life.

7.3 Future Work

There are many possible ways to advance this research. One of them is the evaluation regarding the formulation of relationships explores if different formulations of the same relationship concept lead to different performances. One way would be to use synonyms for the predicates in the factual triples.

Another possibility besides checking whether the PLMs can act as knowledge bases would be to explore whether the PLMs understand temporal inference based on TCBQ. For example, by interrogating two temporal facts and assigning the model, the task of answering the temporal distance between these two events.

To return to the previous point, it would be possible to extend the QG to generate these complex questions automatically. Besides the extension of the QG, the extension of the TCBQ data set would be beneficial.

Conclusion

The main task of this work was to evaluate to what extent PLMs queried by TCBQ are suitable as temporal knowledge bases and if they can be leveraged for KG expansion. To answer this question and summarise the results, each research question will be addressed first.

How much does the TCBQ formulation affect the performance of PLMs?

Concerning the different wording of the questions that were marked with different versions, a difference in performance was always seen. Especially a trend became apparent, namely that questions which were equipped with a time indicator and which must be answered with a single year have an improvement in performance. In section 6.1.1 T0_3B Regarding QV1 - 4, a general improvement in performance was shown by the time indicator, except for When to When and Duration questions. On the other hand, as pointed out in Section 6.1.1 T0_3B Regarding QV1 - 4, when the predicate is lemmatised, it is not possible to say which variant leads to better or worse performance because, in most cases, the lemmatisation had a negligible influence in both directions except for the Yes/No Questions where non-lemmatisation achieved significantly better results. In addition, it also became apparent that the model understood certain subject-object relationships better than others. This insight is described in section 6.1.3 Grouped by Predicate T0_3B. Altogether, it can be concluded that formulation plays an essential role in the performance of PLMs. However, it cannot yet be clearly stated which formulation is best.

How robust are PLMs when given Yes/No questions with correct and incorrect years? Taking into account the data sets QV1R and QV2R, a correctness rate of over 50% can be determined, which at least already gives the model a better performance than if one would flip a coin. However, with a performance of less than one per cent regarding the correct answers of an entity, no robust behaviour can be attributed to the T0_3B model.

What is the performance of PLMs regarding temporal closed-book questions?

By questioning the two models T0_3B and T0pp, generally, no exceptionally good performance was found. Apart from the Yes/No questions, none of the question types achieved a correctness rate of 10%. However, it can be seen from the validity rate that the models at least understood the concept of the questions.

The PLMs used in this evaluation that were surveyed using TCBQ do not demonstrate quality to act as a temporal knowledge base. Taking into account all the possible improvements suggested in the section 7.1 Discussion and the fact that T0_3B was able to solve each task in a rudimentary way and an improvement in performance by using T0pp could be shown (excluding Yes/No questions), it cannot be excluded that future larger models can act as temporal knowledge bases. However, with the current performance, an expansion of a KG to a TKG by leveraging PLMs is not possible based on the findings of this work.

List of Figures

4.1	Experiment framework	12
-----	--------------------------------	----

List of Tables

4.1	Closed Book Question Templates	13
4.2	Time indicated Closed Book Question Templates	13
4.3	Valid Answers: Used Functions and Examples for each Question Type	17
4.4	Correct Answers: Used Extraction and Comparison Functions for each Question Type	18
5.1	Summary - YAGO11k and Cleansed YAGO11k	21
5.2	Predicate Summary - YAGO11k and Cleansed YAGO11k	21
5.3	Question Versions	22
5.4	Question listing and Abbreviations	22
6.1	Evaluation of T0_3B on QV1-4	26
6.2	Comparison of T0pp to T0_3B on QV3	27
6.3	Valid Answer percentage - QV1-4 Grouped by Predicate - T0_3B	29
6.4	Correct Answer percentage - QV1-4 Grouped by Predicate - T0_3B	30
6.5	Valid Answer percentage - QV3 Grouped by Predicate - T0pp vs T3_3B	31
6.6	Correct percentage - QV3 Grouped by Predicate - T0pp vs T3_3B	31
6.7	Summary Yes/No Robust - T0_3B	32
6.8	Google Colab Pro+ Hardware Specifications	33
6.9	Processing Time per Question Set	33
1	Summary Grouped by Predicate V1 - T0_3B	51
2	Summary Grouped by Predicate V2 - T0_3B	52
3	Summary Grouped by Predicate V3 - T0_3B	53

4	Summary Grouped by Predicate V4 - T0_3B	54
5	Summary Grouped by Predicate V3 - T0pp	55

List of Code

8.1	Normalisation of the Answer	57
8.2	Is Yes/No Validity Check	57
8.3	Has Year Validity Check	57
8.4	Has Two Years Validation and Extraction	58
8.5	Has Duration Validity Check	58
8.6	Duration Extraction	59
8.7	Year Extraction	59
8.8	Yes/No Correctness Check	59
8.9	Numbers Equal Correctness Check	59
8.10	First and Last Years Equal Correctness Check	60
8.11	Is Year in Interval Correctness Check	60

Acronyms

C4 Colossal Clean Crawled Corpus. 15

GRU Gated Recurrent Units. 1, 5

KG Knowledge Graph. 1–3, 7, 8, 11, 12, 39, 40

LM Language Model. 1, 10

LSTM Long Short-Term Memory. 1, 5

ML Machine Learning. 6

NLP Natural Language Processing. 1, 2, 5–7, 9, 15

PLM Pre-trained Language Model. 1–3, 9–11, 13, 14, 19, 21, 22, 32, 35–37, 39, 40

QG TKGQuestionGenerator. 3, 11–13, 19, 22, 23, 37

RDF Resource Description Framework. 1, 7

RNN Recurrent Neural Network. 1, 5

SSL Self Supervised Learning. 2, 6

TCBQ Temporal Closed-Book Question. 3, 11–14, 16, 21, 37, 39, 40

TKG Temporal Knowledge Graph. 1–3, 8, 11, 12, 19, 40

Bibliography

- Bouziane, Abdelghani, Djelloul Bouchiha, Nouredine Doumi, and Mimoun Malki (2015). „Question Answering Systems: Survey and Trends“. In: *Procedia Computer Science* 73, pp. 366–375. DOI: 10.1016/j.procs.2015.12.005.
- Bowman, Samuel R., Luke Vilnis, Oriol Vinyals, Andrew M. Dai, Rafal Jozefowicz, and Samy Bengio (Nov. 2015). „Generating Sentences from a Continuous Space“. In: *SIGNLL Conference on Computational Natural Language Learning (CONLL)*. arXiv: 1511.06349 [cs.LG].
- Brown, Tom B., Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei (2020). „Language Models are Few-Shot Learners“. In: *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*. Ed. by Hugo Larochelle, Marc’Aurelio Ranzato, Raia Hadsell, Maria-Florina Balcan, and Hsuan-Tien Lin. URL: <https://proceedings.neurips.cc/paper/2020/hash/1457c0d6bfc4967418bfb8ac142f64a-Abstract.html>.
- Chen, Danqi, Adam Fisch, Jason Weston, and Antoine Bordes (2017). „Reading Wikipedia to Answer Open-Domain Questions“. In: *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*. Ed. by Regina Barzilay and Min-Yen Kan. Association for Computational Linguistics, pp. 1870–1879. DOI: 10.18653/v1/P17-1171. URL: <https://doi.org/10.18653/v1/P17-1171>.
- Cho, Kyunghyun, Bart van Merriënboer, Çaglar Gülçehre, Fethi Bougares, Holger Schwenk, and Yoshua Bengio (2014). „Learning Phrase Representations using RNN Encoder-Decoder for Statistical Machine Translation“. In: *CoRR* abs/1406.1078. arXiv: 1406.1078. URL: <http://arxiv.org/abs/1406.1078>.
- Chung, Junyoung, Çaglar Gülçehre, KyungHyun Cho, and Yoshua Bengio (2014). „Empirical Evaluation of Gated Recurrent Neural Networks on Sequence Modeling“. In: *CoRR* abs/1412.3555. arXiv: 1412.3555. URL: <http://arxiv.org/abs/1412.3555>.

- Clark, Christopher, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova (2019). „BoolQ: Exploring the Surprising Difficulty of Natural Yes/No Questions“. In: *CoRR* abs/1905.10044. arXiv: 1905.10044. URL: <http://arxiv.org/abs/1905.10044>.
- Dasgupta, Shib Sankar, Swayambhu Nath Ray, and Partha Talukdar (2018). „HyTE: Hyperplane-based Temporally aware Knowledge Graph Embedding“. In: *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*. Brussels, Belgium: Association for Computational Linguistics, pp. 2001–2011. DOI: 10.18653/v1/D18-1225. URL: <https://aclanthology.org/D18-1225>.
- Devlin, Jacob, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova (2019). „BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding“. In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*. Ed. by Jill Burstein, Christy Doran, and Tamar Solorio. Association for Computational Linguistics, pp. 4171–4186. DOI: 10.18653/v1/n19-1423. URL: <https://doi.org/10.18653/v1/n19-1423>.
- Dhingra, Bhuwan, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen (June 2021). „Time-Aware Language Models as Temporal Knowledge Bases“. In: *Transactions of the Association for Computational Linguistics 2022; 10 257-273*. DOI: 10.1162/tac1_a_00459. arXiv: 2106.15110 [cs.CL].
- Dong, Li, Nan Yang, Wenhui Wang, Furu Wei, Xiaodong Liu, Yu Wang, Jianfeng Gao, Ming Zhou, and Hsiao-Wuen Hon (2019). „Unified Language Model Pre-training for Natural Language Understanding and Generation“. In: *Advances in Neural Information Processing Systems* abs/1905.03197. arXiv: 1905.03197. URL: <http://arxiv.org/abs/1905.03197>.
- Dong, Xin Luna, Evgeniy Gabrilovich, Jeremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmamm, Shaohua Sun, and Wei Zhang (2014). „Knowledge Vault: A Web-Scale Approach to Probabilistic Knowledge Fusion“. In: *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*. Evgeniy Gabrilovich Wilko Horn Ni Lao Kevin Murphy Thomas Strohmamm Shaohua Sun Wei Zhang Jeremy Heitz, pp. 601–610. URL: <http://www.cs.cmu.edu/~nlao/publication/2014.kdd.pdf>.
- Ehrlinger, Lisa and Wolfram Wöß (2016). „Towards a definition of knowledge graphs.“ In: *SEMANTiCS (Posters, Demos, SuCCESS)* 48.1-4, p. 2.
- Erhan, Dumitru, Yoshua Bengio, Aaron Courville, Pierre-Antoine Manzagol, Pascal Vincent, and Samy Bengio (2010). „Why Does Unsupervised Pre-Training Help Deep Learning?“ In: *J. Mach. Learn. Res.* 11, pp. 625–660. ISSN: 1532-4435.
- Hestness, Joel, Sharan Narang, Newsha Ardalani, Gregory F. Diamos, Heewoo Jun, Hassan Kianinejad, Md. Mostofa Ali Patwary, Yang Yang, and Yanqi Zhou (2017). „Deep Learning Scaling is Predictable, Empirically“. In: *CoRR* abs/1712.00409. arXiv: 1712.00409. URL: <http://arxiv.org/abs/1712.00409>.

- Hochreiter, Sepp and Jürgen Schmidhuber (1997). „Long Short-Term Memory“. In: *Neural Computation* 9.8, pp. 1735–1780. DOI: 10.1162/neco.1997.9.8.1735.
- Hogan, Aidan, Eva Blomqvist, Michael Cochez, Claudia d’Amato, Gerard de Melo, Claudio Gutierrez, José Emilio Labra Gayo, Sabrina Kirrane, Sebastian Neumaier, Axel Polleres, Roberto Navigli, Axel-Cyrille Ngonga Ngomo, Sabbir M. Rashid, Anisa Rula, Lukas Schmelzeisen, Juan Sequeda, Steffen Staab, and Antoine Zimmermann (Mar. 2020). „Knowledge Graphs“. In: *ACM Comput. Surv.* 54(4): 71:1-71:37 (2021). DOI: 10.1145/3447772. arXiv: 2003.02320 [cs.AI].
- Ji, Shaoxiong, Shirui Pan, Erik Cambria, Pekka Marttinen, and Philip S. Yu (Feb. 2020). „A Survey on Knowledge Graphs: Representation, Acquisition and Applications“. In: *IEEE Transactions on Neural Networks and Learning Systems*, 2021. DOI: 10.1109/TNNLS.2021.3070843. arXiv: 2002.00388 [cs.CL].
- Kolen, John F. and Stefan C. Kremer (2001). „Gradient Flow in Recurrent Nets: The Difficulty of Learning LongTerm Dependencies“. In: *A Field Guide to Dynamical Recurrent Networks*, pp. 237–243. DOI: 10.1109/9780470544037.ch14.
- Lan, Zhenzhong, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut (2019). „ALBERT: A Lite BERT for Self-supervised Learning of Language Representations“. In: *International Conference on Learning Representations* abs/1909.11942. arXiv: 1909.11942. URL: <http://arxiv.org/abs/1909.11942>.
- Mahdisoltani, Farzaneh, Joanna Biega, and Fabian M. Suchanek (2015). „YAGO3: A Knowledge Base from Multilingual Wikipedias“. In: *Seventh Biennial Conference on Innovative Data Systems Research, CIDR 2015, Asilomar, CA, USA, January 4-7, 2015, Online Proceedings*. www.cidrdb.org. URL: http://cidrdb.org/cidr2015/Papers/CIDR15%5C_Paper1.pdf.
- McCann, Bryan, Nitish Shirish Keskar, Caiming Xiong, and Richard Socher (2018). „The Natural Language Decathlon: Multitask Learning as Question Answering“. In: *CoRR* abs/1806.08730. arXiv: 1806.08730. URL: <http://arxiv.org/abs/1806.08730>.
- Nickel, Maximilian, Kevin Murphy, Volker Tresp, and Evgeniy Gabrilovich (2015). „A review of relational machine learning for knowledge graphs“. In: *Proceedings of the IEEE* 104.1, pp. 11–33.
- Otter, Daniel W., Julian R. Medina, and Jugal K. Kalita (2021). „A Survey of the Usages of Deep Learning for Natural Language Processing“. In: *IEEE Transactions on Neural Networks and Learning Systems* 32.2, pp. 604–624. DOI: 10.1109/TNNLS.2020.2979670.
- Pan, Sinno Jialin and Qiang Yang (2010). „A Survey on Transfer Learning“. In: *IEEE Transactions on Knowledge and Data Engineering* 22.10, pp. 1345–1359. DOI: 10.1109/tkde.2009.191.
- Peters, Matthew E., Sebastian Ruder, and Noah A. Smith (2019). „To Tune or Not to Tune? Adapting Pretrained Representations to Diverse Tasks“. In: *CoRR* abs/1903.05987. arXiv: 1903.05987. URL: <http://arxiv.org/abs/1903.05987>.

- Petroni, Fabio, Tim Rocktäschel, Patrick S. H. Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H. Miller, and Sebastian Riedel (2019). „Language Models as Knowledge Bases?“ In: *CoRR* abs/1909.01066. arXiv: 1909.01066. URL: <http://arxiv.org/abs/1909.01066>.
- Plisson, Joël, Nada Lavrac, Dunja Mladenic, et al. (2004). „A rule based approach to word lemmatization“. In: *Proceedings of IS*. Vol. 3, pp. 83–86.
- Pörner, Nina, Ulli Waltinger, and Hinrich Schütze (2019). „BERT is Not a Knowledge Base (Yet): Factual Knowledge vs. Name-Based Reasoning in Unsupervised QA“. In: *CoRR* abs/1911.03681. arXiv: 1911.03681. URL: <http://arxiv.org/abs/1911.03681>.
- Qiu, Xipeng, Tianxiang Sun, Yige Xu, Yunfan Shao, Ning Dai, and Xuanjing Huang (Mar. 2020). „Pre-trained Models for Natural Language Processing: A Survey“. In: *SCIENCE CHINA Technological Sciences*, 2020, 63, 1872-1897. DOI: 10.1007/s11431-020-1647-3. arXiv: 2003.08271 [cs.CL].
- Radford, Alec, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. (2019). „Language models are unsupervised multitask learners“. In: *OpenAI blog* 1.8, p. 9.
- Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu (2019). „Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer“. In: *J. Mach. Learn. Res.* abs/1910.10683. arXiv: 1910.10683. URL: <http://arxiv.org/abs/1910.10683>.
- Rajpurkar, Pranav, Jian Zhang, Konstantin Lopyrev, and Percy Liang (2016). „SQuAD: 100, 000+ Questions for Machine Comprehension of Text“. In: *CoRR* abs/1606.05250. arXiv: 1606.05250. URL: <http://arxiv.org/abs/1606.05250>.
- Roberts, Adam, Colin Raffel, and Noam Shazeer (2020). „How Much Knowledge Can You Pack Into the Parameters of a Language Model?“ In: *CoRR* abs/2002.08910. arXiv: 2002.08910. URL: <https://arxiv.org/abs/2002.08910>.
- Sanh, Victor, Albert Webson, Colin Raffel, Stephen H. Bach, Lintang Sutawika, Zaid Alyafeai, Antoine Chaffin, Arnaud Stiegler, Teven Le Scao, Arun Raja, Manan Dey, M. Saiful Bari, Canwen Xu, Urmish Thakker, Shanya Sharma, Eliza Szczechla, Taewoon Kim, Gunjan Chhablani, Nihal V. Nayak, Debajyoti Datta, Jonathan Chang, Mike Tian-Jian Jiang, Han Wang, Matteo Manica, Sheng Shen, Zheng Xin Yong, Harshit Pandey, Rachel Bawden, Thomas Wang, Trishala Neeraj, Jos Rozen, Abheesht Sharma, Andrea Santilli, Thibault Févry, Jason Alan Fries, Ryan Teehan, Stella Biderman, Leo Gao, Tali Bers, Thomas Wolf, and Alexander M. Rush (2021). „Multitask Prompted Training Enables Zero-Shot Task Generalization“. In: *CoRR* abs/2110.08207. arXiv: 2110.08207. URL: <https://arxiv.org/abs/2110.08207>.
- Saxena, Apoorv, Soumen Chakrabarti, and Partha P. Talukdar (2021). „Question Answering Over Temporal Knowledge Graphs“. In: *CoRR* abs/2106.01515. arXiv: 2106.01515. URL: <https://arxiv.org/abs/2106.01515>.

- Suchanek, Fabian, Gjergji M Kasneci, and Gerhard M Weikum (May 2007). „Yago: A Core of Semantic Knowledge Unifying WordNet and Wikipedia“. In: *16th international conference on World Wide Web*. Proceedings of the 16th international conference on World Wide Web. Banff, Canada, pp. 697–697. DOI: 10.1145/1242572.1242667. URL: <https://hal.archives-ouvertes.fr/hal-01472497>.
- Sutskever, Ilya, Oriol Vinyals, and Quoc V. Le (2014). „Sequence to Sequence Learning with Neural Networks“. In: *CoRR* abs/1409.3215. arXiv: 1409.3215. URL: <http://arxiv.org/abs/1409.3215>.
- Taylor, Wilson L. (1953). „Cloze Procedure”: A New Tool for Measuring Readability“. In: *Journalism Quarterly* 30.4, pp. 415–433. DOI: 10.1177/107769905303000401. eprint: <https://doi.org/10.1177/107769905303000401>. URL: <https://doi.org/10.1177/107769905303000401>.
- Vaswani, Ashish, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, and Illia Polosukhin (2017). „Attention Is All You Need“. In: *Advances in neural information processing systems* abs/1706.03762. arXiv: 1706.03762. URL: <http://arxiv.org/abs/1706.03762>.
- Wang, Alex, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman (2018). „GLUE: A Multi-Task Benchmark and Analysis Platform for Natural Language Understanding“. In: *CoRR* abs/1804.07461. arXiv: 1804.07461. URL: <http://arxiv.org/abs/1804.07461>.
- Yang, Liu, Steve Hanneke, and Jaime Carbonell (2013). „A theory of transfer learning with applications to active learning“. In: *Machine learning* 90.2, pp. 161–189.
- Zhang, Sheng, Xiaodong Liu, Jingjing Liu, Jianfeng Gao, Kevin Duh, and Benjamin Van Durme (2018). „ReCoRD: Bridging the Gap between Human and Machine Commonsense Reading Comprehension“. In: *CoRR* abs/1810.12885. arXiv: 1810.12885. URL: <http://arxiv.org/abs/1810.12885>.
- Zhu, Fengbin, Wenqiang Lei, Chao Wang, Jianming Zheng, Soujanya Poria, and Tat-Seng Chua (2021). „Retrieving and Reading: A Comprehensive Survey on Open-domain Question Answering“. In: *CoRR* abs/2101.00774. arXiv: 2101.00774. URL: <https://arxiv.org/abs/2101.00774>.

Appendix - Summary Grouped by Predicate

	Yes/No V1	When V1	Until V1	Left Open V1	Right Open V1	When to When V1	Duration V1
works at valid answer	209	209	209	209	137	91	189
works at size	209	209	209	209	209	209	209
works at correct answer	10	25	5	1	1	0	3
works at valid answer %	100.0	100.0	100.0	100.0	65.55	43.54	90.42
works at correct answer %	4.78	11.95	2.39	0.48	0.48	0.0	1.44
was born in valid answer	2641.0	3340.0					
was born in size	3340.0	3340.0					
was born in correct answer	684.0	87.0					
was born in valid answer %	79.07	100.0					
was born in correct answer %	20.48	2.6					
plays for valid answer	3073	3269	3213	3306	1893	374	2967
plays for size	3312	3312	3308	3308	3308	3308	3308
plays for correct answer	822	530	172	78	171	6	265
plays for valid answer %	92.78	98.7	97.13	99.94	57.22	11.31	89.69
plays for correct answer %	24.82	16.0	5.2	2.36	5.17	0.18	8.01
owns valid answer	329	364	344	361	262	64	208
owns size	364	364	361	361	361	361	361
owns correct answer	21	80	3	0	6	0	1
owns valid answer %	90.38	100.0	95.28	100.0	72.58	17.73	57.62
owns correct answer %	5.77	21.98	0.83	0.0	1.66	0.0	0.27
is married to valid answer	1160	1364	1352	1354	1058	15	1046
is married to size	1377	1377	1363	1363	1363	1363	1363
is married to correct answer	172	193	23	14	14	0	22
is married to valid answer %	84.24	99.06	99.19	99.33	77.62	1.09	76.74
is married to correct answer %	12.49	14.02	1.69	1.03	1.03	0.0	1.60
is affiliated to valid answer	620	627	593	609	311	373	409
is affiliated to size	631	631	627	627	627	627	627
is affiliated to correct answer	5	73	9	19	0	0	3
is affiliated to valid answer %	98.26	99.37	94.58	97.13	49.6	59.48	65.23
is affiliated to correct answer %	0.79	11.57	1.44	3.03	0.0	0.0	0.48
has won prize valid answer	3247	3306	102	102	88	1	91
has won prize size	3307	3307	102	102	102	102	102
has won prize correct answer	1863	72	5	0	5	0	5
has won prize valid answer %	98.19	99.97	100.0	100.0	86.27	0.98	89.22
has won prize correct answer %	56.34	2.18	4.9	0.0	4.9	0.0	4.9
graduated from valid answer	623	630	93	93	93	0	93
graduated from size	630	630	93	93	93	93	93
graduated from correct answer	169	11	0	2	2	0	2
graduated from valid answer %	98.89	100.0	100.0	100.0	100.0	0.0	100.0
graduated from correct answer %	26.83	1.75	0.0	2.15	2.15	0.0	2.15
died in valid answer	1631.0	1643.0					
died in size	1643.0	1643.0					
died in correct answer	153.0	31.0					
died in valid answer %	99.27	100.0					
died in correct answer %	9.31	1.89					
created valid answer	1840	1904	423	429	391	2	186
created size	1921	1921	429	429	429	429	429
created correct answer	637	183	15	10	16	0	6
created valid answer %	95.78	99.11	98.6	100.0	91.14	0.47	43.36
created correct answer %	33.16	9.53	3.5	2.33	3.73	0.0	1.4

Table 1: Summary Grouped by Predicate V1 - T0_3B

	Yes/No V2	When V2	Until V2	Left Open V2	Right Open V2	When to When V2	Duration V2
works at valid answer	209	209	209	208	156	199	176
works at size	209	209	209	209	209	209	209
works at correct answer	5	28	3	1	0	0	2
works at valid answer %	100.0	100.0	100.0	99.52	74.64	95.22	84.21
works at correct answer %	2.39	13.4	1.44	0.48	0.0	0.0	0.96
was born in valid answer	2641.0	3340.0					
was born in size	3340.0	3340.0					
was born in correct answer	684.0	87.0					
was born in valid answer %	79.07	100.0					
was born in correct answer %	20.48	2.6					
plays for valid answer	2948	3261	3166	3295	2220	551	2883
plays for size	3312	3312	3308	3308	3308	3308	3308
plays for correct answer	480	510	177	74	202	8	184
plays for valid answer %	89.01	98.46	95.71	99.61	67.11	16.66	87.15
plays for correct answer %	14.49	15.4	5.35	2.23	6.11	0.24	5.56
owns valid answer	327	364	339	361	285	89	213
owns size	364	364	361	361	361	361	361
owns correct answer	13	78	5	0	6	0	1
owns valid answer %	89.84	100.0	93.91	100.0	78.95	24.65	59.0
owns correct answer %	3.57	21.43	1.39	0.0	1.66	0.0	0.27
is married to valid answer	1160	1364	1352	1354	1058	15	1046
is married to size	1377	1377	1363	1363	1363	1363	1363
is married to correct answer	172	193	23	14	14	0	22
is married to valid answer %	84.24	99.06	99.19	99.33	77.62	1.09	76.74
is married to correct answer %	12.49	14.02	1.69	1.03	1.03	0.0	1.6
is affiliated to valid answer	620	627	593	609	311	373	409
is affiliated to size	631	631	627	627	627	627	627
is affiliated to correct answer	5	73	9	19	0	0	3
is affiliated to valid answer %	98.26	99.37	94.58	97.13	49.6	59.48	65.23
is affiliated to correct answer %	0.79	11.57	1.44	3.03	0.0	0.0	0.48
has won prize valid answer	3073	3305	102	102	100	1	77
has won prize size	3307	3307	102	102	102	102	102
has won prize correct answer	1690	74	6	3	5	0	3
has won prize valid answer %	92.92	99.94	100.0	100.0	98.04	0.98	75.49
has won prize correct answer %	51.1	2.23	5.88	2.94	4.9	0.0	2.94
graduated from valid answer	611	630	93	93	93	0	91
graduated from size	630	630	93	93	93	93	93
graduated from correct answer	93	7	0	3	2	0	3
graduated from valid answer %	96.98	100.0	100.0	100.0	100.0	0.0	97.85
graduated from correct answer %	14.76	1.11	0.0	3.23	2.15	0.0	3.23
died in valid answer	1617.0	1643.0					
died in size	1643.0	1643.0					
died in correct answer	35.0	37.0					
died in valid answer %	98.42	100.0					
died in correct answer %	2.13	2.25					
created valid answer	1787	1904	419	429	399	5	158
created size	1921	1921	429	429	429	429	429
created correct answer	359	192	12	10	17	0	5
created valid answer %	93.02	99.11	97.67	100.0	93.01	1.17	36.83
created correct answer %	18.69	9.99	2.8	2.33	3.96	0.0	1.17

Table 2: Summary Grouped by Predicate V2 - T0_3B

	Yes/No V3	When V3	Until V3	Left Open V3	Right Open V3	When to When V3	Duration V3
works at valid answer	209	209	209	209	209	0	209
works at size	209	209	209	209	209	209	209
works at correct answer	37	31	5	3	4	0	3
works at valid answer %	100.0	100.0	100.0	100.0	100.0	0.0	100.0
works at correct answer %	17.7	14.82	2.39	1.44	1.91	0.0	1.44
was born in valid answer	2331.0	3340.0					
was born in size	3340.0	3340.0					
was born in correct answer	1595.0	62.0					
was born in valid answer %	69.78	100.0					
was born in correct answer %	47.75	1.85					
plays for valid answer	3114	3312	3308	3308	3308	0	3290
plays for size	3312	3312	3308	3308	3308	3308	3308
plays for correct answer	1286	577	201	200	335	0	38
plays for valid answer %	94.02	100.0	100.0	100.0	100.0	0.0	99.46
plays for correct answer %	38.83	17.41	6.08	6.05	10.13	0.0	1.15
owns valid answer	342	364	361	361	361	0	353
owns size	364	364	361	361	361	361	361
owns correct answer	50	77	8	2	11	0	4
owns valid answer %	93.96	100.0	100.0	100.0	100.0	0.0	97.78
owns correct answer %	13.74	21.15	2.22	0.54	3.05	0.0	1.11
is married to valid answer	1243	1377	1363	1363	1363	0	1359
is married to size	1377	1377	1363	1363	1363	1363	1363
is married to correct answer	476	288	24	12	32	0	58
is married to valid answer %	90.27	100.0	100.0	100.0	100.0	0.0	99.71
is married to correct answer %	34.57	20.91	1.76	0.88	2.35	0.0	4.26
is affiliated to valid answer	623	631	627	627	627	0	626
is affiliated to size	631	631	627	627	627	627	627
is affiliated to correct answer	8	93	10	19	15	0	7
is affiliated to valid answer %	98.72	100.0	100.0	100.0	100.0	0.0	99.83
is affiliated to correct answer %	1.27	14.74	1.59	3.03	2.39	0.0	1.11
has won prize valid answer	3286	3307	102	102	102	0	102
has won prize size	3307	3307	102	102	102	102	102
has won prize correct answer	2608	79	1	5	6	0	8
has won prize valid answer %	99.36	100.0	100.0	100.0	100.0	0.0	100.0
has won prize correct answer %	78.86	2.39	0.98	4.9	5.88	0.0	7.84
graduated from valid answer	594	630	93	93	93	0	93
graduated from size	630	630	93	93	93	93	93
graduated from correct answer	340	12	1	3	1	0	10
graduated from valid answer %	94.28	100.0	100.0	100.0	100.0	0.0	100.0
graduated from correct answer %	53.97	1.9	1.08	3.23	1.08	0.0	10.75
died in valid answer	1634.0	1643.0					
died in size	1643.0	1643.0					
died in correct answer	778.0	20.0					
died in valid answer %	99.45	100.0					
died in correct answer %	47.34	1.22					
created valid answer	1852	1918	426	429	429	0	419
created size	1921	1921	429	429	429	429	429
created correct answer	919	198	14	12	18	0	12
created valid answer %	96.41	99.83	99.3	100.0	100.0	0.0	97.67
created correct answer %	47.83	10.31	3.26	2.80	4.2	0.0	2.80

Table 3: Summary Grouped by Predicate V3 - T0_3B

	Yes/No V4	When V4	Until V4	Left Open V4	Right Open V4	When to When V4	Duration V4
works at valid answer	209	209	209	209	209	0	209
works at size	209	209	209	209	209	209	209
works at correct answer	16	32	5	2	3	0	2
works at valid answer %	100.0	100.0	100.0	100.0	100.0	0.0	100.0
works at correct answer %	7.66	15.31	2.39	0.96	1.44	0.0	0.96
was born in valid answer	2331.0	3340.0					
was born in size	3340.0	3340.0					
was born in correct answer	1595.0	62.0					
was born in valid answer %	69.78	100.0					
was born in correct answer %	47.75	1.85					
plays for valid answer	2992	3312	3308	3308	3308	0	3290
plays for size	3312	3312	3308	3308	3308	3308	3308
plays for correct answer	1038	583	193	216	352	0	38
plays for valid answer %	90.34	100.0	100.0	100.0	100.0	0.0	99.46
plays for correct answer %	31.34	17.59	5.83	6.52	10.63	0.0	1.15
owns valid answer	338	364	361	361	361	0	352
owns size	364	364	361	361	361	361	361
owns correct answer	33	77	6	3	11	0	4
owns valid answer %	92.86	100.0	100.0	100.0	100.0	0.0	97.5
owns correct answer %	9.07	21.15	1.66	0.83	3.05	0.0	1.11
is married to valid answer	1243	1377	1363	1363	1363	0	1359
is married to size	1377	1377	1363	1363	1363	1363	1363
is married to correct answer	476	288	24	12	32	0	58
is married to valid answer %	90.27	100.0	100.0	100.0	100.0	0.0	99.71
is married to correct answer %	34.57	20.91	1.76	0.88	2.35	0.0	4.26
is affiliated to valid answer	623	631	627	627	627	0	626
is affiliated to size	631	631	627	627	627	627	627
is affiliated to correct answer	8	93	10	19	15	0	7
is affiliated to valid answer %	98.72	100.0	100.0	100.0	100.0	0.0	99.83
is affiliated to correct answer %	1.27	14.74	1.59	3.03	2.39	0.0	1.11
has won prize valid answer	3238	3307	102	102	102	0	102
has won prize size	3307	3307	102	102	102	102	102
has won prize correct answer	2372	82	1	6	7	0	8
has won prize valid answer %	97.91	100.0	100.0	100.0	100.0	0.0	100.0
has won prize correct answer %	71.73	2.48	0.98	5.88	6.85	0.0	7.84
graduated from valid answer	581	630	93	93	93	0	93
graduated from size	630	630	93	93	93	93	93
graduated from correct answer	199	13	1	3	1	0	14
graduated from valid answer %	92.22	100.0	100.0	100.0	100.0	0.0	100.0
graduated from correct answer %	31.59	2.06	1.08	3.23	1.08	0.0	15.04
died in valid answer	1628.0	1643.0					
died in size	1643.0	1643.0					
died in correct answer	399.0	19.0					
died in valid answer %	99.09	100.0					
died in correct answer %	24.27	1.16					
created valid answer	1838	1918	426	429	429	0	411
created size	1921	1921	429	429	429	429	429
created correct answer	621	197	13	11	18	0	10
created valid answer %	95.67	99.83	99.3	100.0	100.0	0.0	95.8
created correct answer %	32.33	10.26	3.03	2.56	4.2	0.0	2.33

Table 4: Summary Grouped by Predicate V4 - T0_3B

	Yes/No V3	When V3	Until V3	Left Open V3	Right Open V3	When to When V3	Duration V3
works at valid answer	209	209	209	209	209	203	208
works at size	209	209	209	209	209	209	209
works at correct answer	65	30	6	10	6	0	2
works at valid answer %	100.0	100.0	100.0	100.0	100.0	97.13	99.52
works at correct answer %	31.1	14.35	2.87	4.78	2.87	0.0	0.96
was born in valid answer	3200.0	3340.0					
was born in size	3340.0	3340.0					
was born in correct answer	768.0	89.0					
was born in valid answer %	95.81	100.0					
was born in correct answer %	22.99	2.65					
plays for valid answer	3295	3312	3308	3308	3308	1312	3307
plays for size	3312	3312	3308	3308	3308	3308	3308
plays for correct answer	984	570	152	334	251	29	387
plays for valid answer %	99.49	100.0	100.0	100.0	100.0	39.66	99.97
plays for correct answer %	29.70	17.21	4.59	10.1	7.59	0.88	11.7
owns valid answer	345	364	361	361	361	224	338
owns size	364	364	361	361	361	361	361
owns correct answer	147	82	12	11	17	3	4
owns valid answer %	94.78	100.0	100.0	100.0	100.0	62.05	93.63
owns correct answer %	40.38	22.53	3.32	3.05	4.71	0.83	1.11
is married to valid answer	1377	1377	1363	1363	1363	1239	1363
is married to size	1377	1377	1363	1363	1363	1363	1363
is married to correct answer	240	374	39	51	57	3	41
is married to valid answer %	100.0	100.0	100.0	100.0	100.0	90.9	100.0
is married to correct answer %	17.43	27.16	2.86	3.74	4.18	0.22	3.01
is affiliated to valid answer	630	631	627	627	627	592	626
is affiliated to size	631	631	627	627	627	627	627
is affiliated to correct answer	126	97	17	49	34	2	13
is affiliated to valid answer %	99.83	100.0	100.0	100.0	100.0	94.42	99.83
is affiliated to correct answer %	19.97	15.37	2.71	7.81	5.42	0.32	2.07
has won prize valid answer	3307	3307	102	102	102	102	102
has won prize size	3307	3307	102	102	102	102	102
has won prize correct answer	1794	95	3	4	13	0	5
has won prize valid answer %	100.0	100.0	100.0	100.0	100.0	100.0	100.0
has won prize correct answer %	54.25	2.87	2.94	3.92	12.75	0.0	4.9
graduated from valid answer	630	630	93	93	93	93	90
graduated from size	630	630	93	93	93	93	93
graduated from correct answer	200	12	1	3	6	0	1
graduated from valid answer %	100.0	100.0	100.0	100.0	100.0	100.0	96.77
graduated from correct answer %	31.75	1.9	1.08	3.23	6.45	0.0	1.08
died in valid answer	1638.0	1638.0					
died in size	1643.0	1643.0					
died in correct answer	1519.0	41.0					
died in valid answer %	99.7	99.7					
died in correct answer %	92.45	2.5					
created valid answer	1914	1920	429	429	429	416	428
created size	1921	1921	429	429	429	429	429
created correct answer	672	200	9	28	39	2	9
created valid answer %	99.64	99.95	100.0	100.0	100.0	96.97	99.77
created correct answer %	34.98	10.41	2.1	6.52	9.09	0.47	2.1

Table 5: Summary Grouped by Predicate V3 - T0pp

Appendix - Assessment of the Models' Answers Python Functions

Every listed function is compatible with Python Version 3.8.

```
def remove_model_answer_extensions(model_an: str):
    model_an = str(model_an)
    model_an = model_an[6: len(model_an) - 4]
return model_an
```

```
def lower_case(model_an: str):
    return model_an.lower()
```

Listing 8.1: Normalisation of the Answer

```
def is_yes_no(model_an: str):
    if model_an == 'yes' or model_an == 'no':
        return True
    return False
```

Listing 8.2: Is Yes/No Validity Check

```
import re
```

```
def has_year(string: str):
    m = re.search(r'[1-3][0-9]{3}', string)
    if m:
        return True
    return False
```

Listing 8.3: Has Year Validity Check

```

def extract_two_years(string: str):
    matches = re.findall(r'[1-3][0-9]{3}', string)
    ret = []
    if len(matches) == 2:
        ret.append(matches[0])
        ret.append(matches[1])
        return ret
    ret.append(string)
    return ret

```

```

def has_two_entries(model_an):
    if len(model_an) == 2:
        return True
    return False

```

Listing 8.4: Has Two Years Validation and Extraction

```

def has_duration(model_an: str):
    if has_numbers(model_an):
        return True
    try:
        w2n.word_to_num(model_an)
        return True
    except ValueError as e:
        print(e)
    return False

```

```

def has_numbers(model_an: str):
    m = re.search(r'[0-9]+', model_an)
    if m:
        return True
    return False

```

Listing 8.5: Has Duration Validity Check

```

def extract_duration(model_an: str):
    w2n_num = None
    try:
        w2n_num = w2n.word_to_num(model_an)
        return w2n_num
    except ValueError as e:
        print(e)
    if not w2n_num:
        return int(extract_number(model_an))
    return model_an

```

Listing 8.6: Duration Extraction

```

import re

def extract_year(string: str):
    m = re.search(r'[1-3][0-9]{3}', string)
    if m:
        year = m.group(0)
        return year
    m = re.search(r'[0-9]{3}', string)
    if m:
        year = m.group(0)
        return year
    return string

```

Listing 8.7: Year Extraction

```

def is_yes_no_correct(an, model_an):
    if an == model_an:
        return True
    return False

```

Listing 8.8: Yes/No Correctness Check

```

def is_equal(model_an, an, valid_answer):
    if valid_answer:
        if an:
            an = int(an)
        if an == model_an:
            return True
    return False

```

Listing 8.9: Numbers Equal Correctness Check

```

def is_first_and_last_equal(model_an, an, valid_answer):
    if valid_answer:
        an_first = int(an[0])
        an_last = int(an[1])

        model_an_first = model_an[0]
        model_an_last = model_an[len(model_an) - 1]
        if an_first == model_an_first:
            if an_last == model_an_last:
                return True
    return False

```

```

def extract_number(model_an):
    m = re.search(r'[0-9]+', model_an)
    if m:
        year = m.group(0)
        return year

```

Listing 8.10: First and Last Years Equal Correctness Check

```

def is_answer_in_interval(model_an, an, valid_answer):
    if valid_answer:
        if an:
            an = int(an)
            if an in model_an:
                return True
    return False

```

Listing 8.11: Is Year in Interval Correctness Check