S-TMRC: INTRODUCING STRUCTURE KNOWLEDGE FOR TEMPORAL MACHINE READING COMPREHENSION

Yu Feng, Entropy Xu, Yu Tang
Duke University

ABSTRACT

As time evolves, many facts will evolve along with it, such as 'the U.S. President', 'Home Team of Lebron James', etc. Understanding the scope and interval of knowledge is an essential task for more generalized machine reading comprehension. However almost none existing MRC method particularly takes time dimension into consideration, while existing temporal KBQA methods have achieved strong performance in complicated temporal question answering. Inspired by temporal KBQA models, we therefore propose a new structured temporal MRC model, S-TMRC. A subgraph retriever is designed to retrieve a temporal question subgraph for each question from the corresponding long document, on which a question reasoner can utilize temporal KBQA methods to reason for the answer. Extensive experiments on the TimeQA Chen et al. (2021) benchmark have shown that our proposed model can improve overall temporal QA performance, efficiency and interpretability.

1 INTRODUCTION

Time is an important dimension in our physical world. Lots of facts can evolve with respect to time. For example, the U.S. President might change every four years. In order to answer the following question- who was the U.S. President after WWII? We should capture the knowledge of different presidents over time in the document, while managing to perform comparison between the time of WWII (1939-1945) and their respective serving time to locate the right answer, Harry S. Truman(served as U.S. President from 1945 to 1953) instead of Barack Obama (served as U.S. President from 2009 to 2017). Therefore, it is important to consider the time dimension and empower the existing QA models to reason over time.

However existing models in machine reading comprehension rarely took time dimension into consideration. Dhingra et al. (2021) proposed a simple modification to pretraining that facilitated the acquisition of temporal knowledge and efficient updates to existing pretrained LMs. They focused on coarse and shorter time range for calculating by whole year only from 2010-2020, while we are facing a much larger time span from approximate 1800 to 2021 which will be much more complicated. Some works have also been done in KBQA for temporal reasoning (Lacroix et al. (2020); Jain et al. (2020); Mavromatis et al. (2021)). They have achieved high performance for complex temporal question answering. However, they reasoned on the structured knowledge base while we are facing pure textual context (i.e., passages).

To take advantage of the strong performance of the temporal KBQA models, we propose S-TMRC, a new structured temporal machine reading comprehension model by introducing and constructing structured knowledge. In particular, our proposed S-TMRC split the temporal question answering process into two modules: subgraph retriever and question reasoner. We utilize the subgraph retriver to construct a temporal subgraph for each temporal MRC question from its corresponding long document and then apply temporal KBQA model as the question reasoner for answer prediction.

Existing MRC models can only predict the location of possible answers, which leads to poor interpretability for the question answering process. The temporal question subgraph retrieved by our subgraph retriever can serve as explicit guidance for users to understand the question answering process. Also, to reason directly on long paragraphs is extremely time and energy consuming, our proposed model can dramatically reduce training time and memory consumption by breaking the reasoning process down into simple and efficient parts.

Experimental results on the TimeQA Chen et al. (2021) benchmark, show that our temporary MRC framework can achieve substantial and consistent improvement in terms of answering temporal questions, especially complicated temporal questions.

2 Related Work

Recent advances in large pre-trained language models (Devlin et al. (2019); Liu et al. (2019)) have been shown to encode an impressive amount of factual information (Petroni et al. (2019); Roberts et al. (2020)). This has inspired a line of research on probing and measuring the capacity of stored factual knowledge in LMs Poerner et al. (2019). While large LMs appear to absorb KB-like information as a preproduct of pretraining, degradation of models has been found when tested on a different time period than their training data Huang & Paul (2018). To address the lackness of temporal reasoning ability of pre-trained LMs, Dhingra et al. (2021) proposed a temporally-scoped pre-trained scheme that tried to the impact of temporal shift on the knowledge encoded by existing LMs. Röttger & Pierrehumbert (2021) proposed a temporal adaptation method, trying to capture event-driven changes in language use in the downstream task. A parallel line of work has explored editing interal parameters of pre-trained LMs given a collection of facts (De Cao et al. (2021); Zhu et al. (2020)). Another line of work explore the benefits of leverage the external symbolic temporal information: Several KBQA datasets (Saxena et al. (2021); Neelam et al. (2022)) have been constructed which leveraged the temporal information resides in large-scale knowledge bases such as Freebase and Wikidata. The KBQA datasets make use of Temporal Knowledge Graphs (Temporal KGs) which provide temporal scopes (e.g., start and end times) on each edge in the KG. The dataset format closely resembles the traditional Knowledge Base augmented question answering task albeit the explicit constraint of the answer being either an KB entity or a time duration.

3 Approach

3.1 Problem definition

We aims to solve the temporal machine reading comprehension questions. A temporal question q is a normal question with time constraint, e.g, from year1 to year2, before Month1 year3 (refer Firgue 1 for a detailed example). A long wikidata document D is given for answer a_q prediction.

3.2 Baseline Method

We introduce the base Temporal KBQA model and the baseline temporal MRC models in this section.

Knowledge Base Question Answering

Our KBQA base model TempoQR will first apply TCompelEx Lacroix et al. (2020) to obtain the entity, relation, and timestamp embeddings for the constructed Temporal Knowledge Base. It exploits TKG embeddings to ground the question to the specific entities and time scope it refers to. Then it reasons for question answering by augmenting the question embeddings with context, entity and time-aware information via three specialized modules. The first computes a textual representation of a given question, the second combines it with the entity embeddings for entities involved in the question, and the third generates question-specific time embeddings. Finally, a transformer-based encoder learns to fuse the generated temporal information with the question representation, which is used for answer predictions.

Machine Reading Comprehension

We adpot two baseline models for Temporal MRC Chen et al. (2021) , the BigBird and the FiD which are known to achieve state-of-the-art performance on the Natural Question Answering Problems.

The BigBird Zaheer et al. (2020) extractive model aims to extract the start and end positions from the given sequence. BigBird exploits sparse attention which can handle sequences of extremely

long length. As a consequence of the capability to handle longer context, BigBird drastically improves performance on various natural language processing tasks such as question answering and summarization.

The FiD Izacard & Grave (2020) is an generative model that aims to generate answer token by token in an auto-regressive fashion. Being an generative models, FiD are especially good at aggregating and combining evidence from multiple passages.

3.3 EXTENSIONS

3.3.1 MODEL

Given the aforementioned KBQA model works extremely well on temporal question answering. We adapt MRC task to KBQA task by developing the structured temporal MRC (**S-TMRC**) model in pursuit of better overall QA performance, efficiency and interpretability. In particular, the S-TMRC model is composed of two modules. The **Subgraph Retriever** module targets at retrieving the question-related temporal subgraph from the long document. The **Question Reasoner** module further reasons on the subgraph to locate the target answer.

SUBGRAPH RETRIEVER

A question-related temporal subgraph $\mathcal{G}$ contains temporal triplets related to the given question. A normal triplet $\{(e, r, e')|e, e' \in \mathcal{E}, r \in \mathcal{R}\}$ includes a head entity, relation and tail entity, while a temporal triplet $\{(e, r, e', t_s, t_e)|e, e' \in \mathcal{E}, r \in \mathcal{R}\ , t_s, t_e \in \mathcal{T}\}$ adds extra corresponding event start time and end time. Figure 1 demonstrated a temporal subgraph for the quetsion, "Which school did Cho Yoon-sun go to from 1984 to 1988?". In order to obtain the accurate question subrgaph, we need to first retrieve normal triplets from the question and its corresponding paragraph, and then predict event time for each triplet to construct final temporal triplets.

Noted that we only construct one-hop subgraph for each question, we first use NER to identify the top entity in the question. We infer the target relation from the question by measuring the cosine similarity between embedding (RoBERTa) of question and relation and choose top K=1. We locate the top-K (K=3) relevant paragraphs for the question by measuring the cosine similarity between embedding of question and paragraph topic word. Finally, we rely on **Entity Retriever** and **Time Retriever** to locate the potential tail entities and their corresponding event time for the top entity-relation pair in the relevant paragraphs.

Entity Retriever. We construct two tail entity retrievers, the first one is WikiData-based where we make full use of the WikiData and the second one is language model-based (LM-based) where we do not rely on the WikiData.

WikiData-based: We first map the top entity to the WikiData Q ID and the relation to the WikiData P ID. We then look for normal triplets in WikiData and link the tails with entities in the paragraphs by the existing tool Falcon 2.0 Sakor et al. (2020).

LM-based: We modify the original temporal question into a multi-answer question by removing its time constraint. For example, the aformentioned quetsion "Which school did Cho Yoon-sun go to from 1984 to 1988?" will be simplified into the question "Which school did Cho Yoon-sun go to ?".

We used BERT Large as our base language model. In particular, we use the mask prediction variant of the BERT Large Model. We start with a pre-trained question answering model from the huggingface transformer library. The pre-trained model is pre-trained on the squad dataset with the loss function of the sum of the cross entropy between the output logits and the one-hot encoding of start/end positions as shown in Equation 1.

$$\mathcal{L} = \textbf{CrossEntropy}(\text{logits}, \text{start}(\text{ans})) + \textbf{CrossEntropy}(\text{logits}, \text{end}(\text{ans})) \tag{1}$$

Then, we start fine-tuning using our own MRC multi-answer data set. Because there are multiple answers in a sample in our modified MRC data set, we need to rewrite the loss function of the original model in order to support multiple answers.

We define our fine-tuning loss function as the average of the cross-entropy loss of each correct answers as shown in Equation 2 where A is the set containing all correct answers.

$$\mathcal{L} = \frac{\sum_{\text{ans} \in A} \textbf{CrossEntropy}(\text{logits}, \text{start}(\text{ans})) + \textbf{CrossEntropy}(\text{logits}, \text{end}(\text{ans}))}{|A|} \tag{2}$$

We also need to make the model being able to output multiple answers at inference time. The model outputs logits for the start position and end position both of the length of the context text. We first perform softmax on the both output logits and then sort them in decreasing order. Then, we get the top k items from the list, and the original start/end positions represented by these k items are the k answers that the model need to output. In reality, we choose $k = 5$ for two reasons: in our data set most of the questions have less than 5 answers. Empirically, the answer our model produce with ranking after $k = 5$ have sub-optimal prediction result.

Time Retriever. Our time retriever is consisted of three different layers: a pre-trained language model as the contextualized encoder; a Graph Attention Network as the global time propagation layer and a 2-layer Feed-forward neural network to predict the event timestamps. We also adopt the 4-tuple representation introduced by the TAC-KBP2011 temporal slot filling task Heng et al. (2011), with each event's temporal representation defined as the 4-tuple $(EarliestStart, LatestStart, EarliestEnd, LatestEnd)$. The increased granularity enabled our model to caputure uncertainty within the language expression for a certain timepoint.From the output of our LM, we can directly retrieve contextualized representation for event triggers and time expressions. We use h_{e_i} to denote the event trigger for event e_i and h_{t_j} for the expression for time t_j. We take the average of event trigger/time expressions across multiple tokens to make sure all h_{e_i} and h_{t_j} are of the same dimensions. We futher used Graph Attention Networks (GAT) to capture types of relations between events and aggregating informations across the temporal augmented graph. Compared to the original GAT proposed in ,we further added relational embedding for edge labels. Following the apporach as the original GAT Veličković et al. (2017), we stacked several layers of GAT on top of the contextualized representations, with each layer defined as follows:

$$a_{ij} = \text{LeakyReLU}(w_f[Wh_{v_i}; Wh_{v_j}; \phi_{r_{i,j}}) \tag{3}$$

$$\alpha_{ij} = \frac{\exp(a_{ij})}{\sum_{k \in \mathcal{N}(i)} \exp(a_{ik})} \tag{4}$$

$$h'_{v_i} = \text{ELU}\left(\sum_{k \in \mathcal{N}(i)} \alpha_{ij} W_a h_{v_k}\right) \tag{5}$$

Where h'_{v_i} is the updated embedding after one layer and $\mathcal{N}(i)$ represents the neighbors of e_i. $\phi_{r_{i,j}}$ is the learnable relation embedding between e_i and e_j. The GAT along with the temporal augmented graphs serve as the intermediate layer between the LM and 2-layer FFNN. After the information propagation with the intermediate GATs, we concatenate h_{e_i} and h_{t_j} and use a 2-layer Feed-forward Nerual Network to estimate the probability of filling t_j as the kth element in event e_i's 4-tuple time representation. In short:

$$p_{i,j,k} = \sigma(w_k \, \text{ReLU}(W[h_{e_i}; h_{t_j}] + b) + b_k) \tag{6}$$

QUESTION REASONER

We adopt the base TempoQR model as our question reasoner. We first apply TCompelEx to obtain the entity, relation, and timestamp embeddings for the constructed subgraphs and then use TempoQR to perform the temporal reasoning for answer prediction.

3.3.2 TRAINING STRATEGIES

We adopt two different training strategies: two-step supervised training to explore the optimal performance of our proposal model; end-to-end training to test the performance where we do not include any additional supervision signals other than the original golden answers for the temporal MRC.

TWO-STEP SUPERVISED TRAINING

We train subgraph retriever and question reasoner as two completely separate modules. We first train the component of subgraph retriever: entity retriever and time retriver. We build additional supervision signals for them by constructing the golden temporal subgraph. We infer the question-related temporal subgraphs after training the entity retriever and time retriever, and utlize the subgraphs to train the question reasoner.

END-TO-END TRAINING

The main idea of end-to-end training is to leverage the feedback from the question reasoner to guide subgraph retriever, while training the question reasoner at the same time. To enable this, we optimize the posterior $p(\mathcal{G}|q, a_q)$ instead of the prior $p(\mathcal{G}|q)$, since the former one contains the additional likelihood $p(a|q, g_k)$ which exactly reflects the feedback from the reasoner. we approximate $p(\mathcal{G}|q, a_q)$ by the sum of the probabilities of each potential subgraph and rewrite the posterior of each subgraph by Bayes' rule **?**, i.e.,

$$p(\mathcal{G}|q, a_q) \approx \sum_{k=1}^{n} p(g_k|q, a_q) \propto \sum_{k=1}^{n} p(a_q|q, g_k)p(g_k|q) \tag{7}$$

where $p(g_k|q)$ is the likelihood of the kth potential subgraph and $p(a_q|q, g_k)$ is the likelihood of the answer a_q given the k-th potential subgraph. The whole objective function for each training instance $(q, a, \mathcal{G})$ is formalized as:

$$\mathcal{L} = \max_{\phi} \log p(a_q|q, \mathcal{G}) + \max_{\theta} \log \sum_{k=1}^{n} p(a_q|q, g_k)p(g_k|q) \tag{8}$$

where ϕ is the parameters of question reasoner and θ is the parameters of subgraph retriever.

4 EXPERIMENTS

4.1 DATASETS

We use the TimeQA dataset Chen et al. (2021) to train our model. The dataset is constructed using the following approach:

1. Time-evolving facts are first mined from WikiData which are also aligned with its wikipedia page.
2. Crowd workers are employed to verify and calibrate the noises.
3. Question answering pairs are generated based on the annotated time-sensitive facts.

The data set is split into the training set, development set and the test set with ratio of around 14:3:3. All the data in the data sets are categorized into "easy" mode and "hard" mode. The hard mode samples requires the model to do temporal reasoning to find the correct answer while the easy mode sample's answer is available in the context. The detail of the size of each data sets and samples of each mode are shown in Table 1.

4.2 MODEL AND TRAINING DETAILS

BASELINE

We adopt the exact same baselines: BigBird and FiD for the baseline MRC model. We test their performance in three settings: directly finetuning on TimeQA dataset, pretraining with other datasets (TriviaQA/ NQ) only, and pretraining with other datasets and then finetune on TimeQA. For our S-TMRC model, we test the performance of two-step training and end-to-end training. Both WikiData-based Entity Retriever and LM-based Entity Retriever are used for two-step training. Only LM-based Entity Retriever are used for end-to-end training.

Split	Mode	#Questions	#Entities	#Relations	#Answerable	#Unanswerable
Train	Easy	14308	3500	70	12532	1776
	Hard	14681	3500	70	12532	2149
Dev	Easy	3021	748	52	2674	347
	Hard	3087	748	52	2674	413
Test	Easy	2997	749	50	2613	384
	Hard	3078	749	50	2613	465

Table 1: The statistics for different splits and modes for the Time QA dataset.

IMPLEMENTATION DETAILS

All of our models are implemented and trained using PyTorch library on one Nvidia A4000 GPU with 16 G Memory. The LM-based model for Entity Retriever is fine tuned with AdamW optimizer of default parameters with learning rate of $1e-6$. The Time Retriever is trained for 100 epochs with a batch size of 256. The Quetsion Reasoner is trained for 50 epochs with a batch size of 20.

4.3 RESULTS

We compare with baseline MRC models and present the results of our proposed S-TMRC models in Table 2. We can conclude that our proposed model does better than the baseline MRC models when there is not any additional knowledge applied into the model. When we include additional knowledge, we can see that the baseline MRC models finetuned on NQ/TriviaQA does better than our optimal S-TMRC model where we use the golden temporal subgraph (constructed by authors) on easy mode, while our S-TMRC model performs better on hard-mode questions. This indicates our model's capability of complicated temporal question answering by explicitly interpreting and reasoning on time. The two-step S-TMRC models have higher performance than end-to-end S-TMRC model as more supervision signals are given in two-step S-TMRC models.

Model	EM(Easy-Mode)	EM(Hard-Mode)
Baseline MRC Models		
BigBird (FT on TimeQA)	16.3	11.9
FiD (FT on TimeQA)	15.7	10.3
BigBird (FT on TriviaQA)	33.7	27.7
FiD (FT on NQ)	23.3	16.0
BigBird (FT on TriviaQA + TimeQA)	50.8	44.4
FiD (FT on NQ + TimeQA)	60.5	46.8
Two-Step S-TMRC Models		
WikiData-based	41.1	42.3
LM-based	39.4	42.0
End-to-End S-TMRC Model		
LM-based	33.6	37.4
Optimal S-TMRC Model	55.8	60.2
Human Worker	89.0	87.0

Table 2: Main results for different models on the TimeQA dataset, we report the EM scores for test set under easy and hard mode.

5 ANALYSIS

5.1 QUALITY OF SUBGRAPH RETRIEVER

We first evaluate the performance of Entity Retriever, whose accuracy is crucial for effective subgraph construction. We evaluate on LM-based Entity Retriever. We vary the proportion of the supervised data in 0%, 25%, 50%, 75%, 100%, and present the corresponding accuracy of tail entities in the subgraph by measuring the answer coverage rate in the modified multi-answer questions

% Training Set used for entity retriever	0%	25%	50%	75%	100%
At Least One Answer in a Sample is Correct	67.48	69.65	71.68	72.36	73.85
All Answer in a Sample is Correct	8.52	8.81	10.29	10.30	11.09

Table 3: Results for ablation study of different proportion of supervision signals for entity retriever.

in Table 3. This table shows that increasing the size of supervised data gives us very marginal performance boost. We estimate such small performance boost is due to the limited size of the modified multi-answer MRC dataset, which is 5 times smaller than the original MRC training dataset. In addition, the absolute accuracy of the model is low. The probability that the model will answer all questions correctly in a given sample is only 11.09% at best. We believe the main cause is that the BERT Large model for mask prediction is not originally pre-trained with multiple answers.

We further evaluate the performance of Time Retriever. Table 4 demonstrates system performance (%) on event time extraction on dev/test set. We can find that involving document-level information from embeddings can improve the system performance. The GAT based time propagation method outperforms other baselines as it explicitly encodes temporal relations. Noted that the optimal possible EM score for the dataset is around 68% as the model cannot infer complicated implicit time from the paragraph.

	EM(dev)	EM(test)
Baseline Extraction Model		
RoBERTa	39.1	37.9
Longformer	42.6	40.4
Temporal Information based Propagation		
GAT	47.6	49.3
Optimal	68.1	68.5

Table 4: System performance (%) on event time extraction on dev/test set.Exact match rate of event time is calculated.

5.2 Explainability of S-TMRC

Figure 1 shows a sample case of S-TMRC's reasoning process and its generated subgraph to assist in explaining the answering process of the question. Given the sample question "Which school did Cho Yoon-sun go to from 1984 to 1988?", the baseline MRC model will consume the whole paragraph as input and output the start position and end position of "seoul national university". While for our S-TMRC model, we will first locate the top entity (highlighted in orange) and relation (highlighted in green) of the given question. We then simplify and answer the question "Which schools did Cho Yoon-sun go to?" (highlighted in blue). Finally we predict the event time for each answer candidate (highlighted in yellow). The explicit structured subgraph enables user to interpret the question answering process more clearly.

5.3 Efficiency of S-TMRC

We evaluate S-TMRC of its computation times and memory consumption. The original MRC models are trained on 4 GPU with 24G memory with a per-GPU-batch-size of 1. Our proposed S-TMRC only requires a GPU with 16G memory. Table 5 presents the summary of training times. We can conclude that despite the multiple components in our proposed model, since we break down the problem into small pieces, the overall training time is much less the original MRC models. The inference time is longer, this is expected as S-TMRC require inference for each different component.

5.4 Remaining Challenges

The general strict accuracy for entity retriever is relatively low as existing language models cannot support multi-answer MRC well. We think it is an interesting research direction. Also, our time retriever cannot deal with implicit time prediction which may require common sense reasoning.

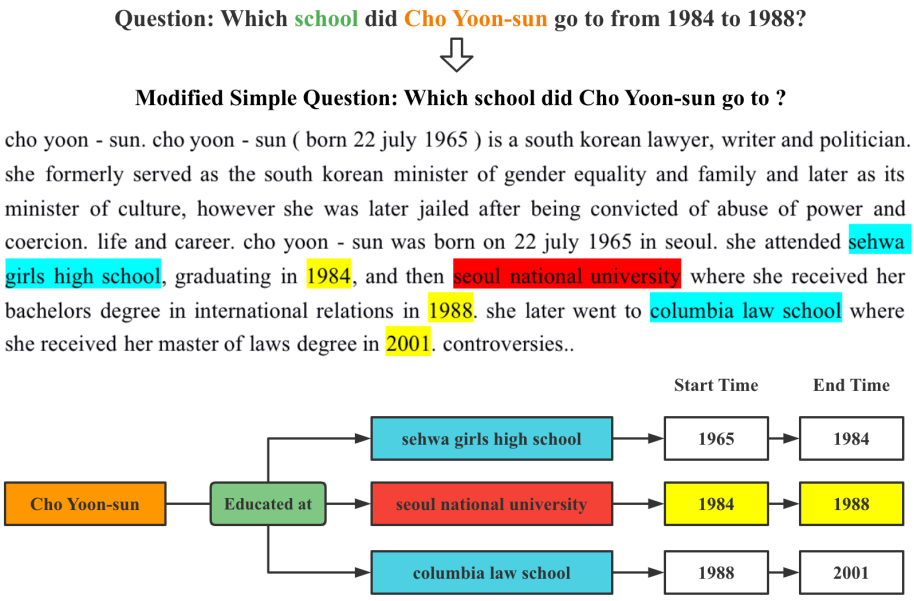

Figure 1: Sample case of S-TMRC model.

	Training Time
Baseline Extraction Model	
BigBird	180 min
FiD	150 min
S-TMRC Models	
Entity Retriever	30 min
Time Retriever	15 min
Question Reasoner	50 min
Total (with data processing)	110 min

Table 5: Training times for different models.

Currently we rely on KB pretrained embeddings for question reasoner. As a result, the model is not new information friendly and require retrain if there is new question including new information. In the future, we can work on a subgraph-based KBQA method which does not require pretrained KB.

6 Conclusion

Reading comprehension with time-evolving facts is an essential problem in the field of natural language processing. However, conventional language models have unsatisfying reasoning performance on temporal knowledge. S-TMRC, which implements a subgraph retriever followed by a question reasoner is proposed in this paper to address the problem. Our evaluation result shows that S-TMRC achieves an accuracy comparable to the baseline models while requires much less training resources, takes shorter amount of time to train and has better interpretability.

REFERENCES

Wenhu Chen, Xinyi Wang, and William Yang Wang. A dataset for answering time-sensitive questions. *arXiv preprint arXiv:2108.06314*, 2021.

Nicola De Cao, Wilker Aziz, and Ivan Titov. Editing factual knowledge in language models. *arXiv preprint arXiv:2104.08164*, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL https://aclanthology.org/N19-1423.

Bhuwan Dhingra, Jeremy R. Cole, Julian Martin Eisenschlos, Daniel Gillick, Jacob Eisenstein, and William W. Cohen. Time-aware language models as temporal knowledge bases, 2021.

Ji Heng, R Grishman, and H Dang. Overview of the tac2011 knowledge base population track. In *Proceedings of the Fourth Text Analysis Conference (TAC 2011)*, 2011.

Xiaolei Huang and Michael J Paul. Examining temporality in document classification. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, 2018.

Gautier Izacard and Edouard Grave. Leveraging passage retrieval with generative models for open domain question answering, 2020. URL https://arxiv.org/abs/2007.01282.

Prachi Jain, Sushant Rathi, Mausam, and Soumen Chakrabarti. Temporal Knowledge Base Completion: New Algorithms and Evaluation Protocols. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 3733–3747, Online, November 2020. Association for Computational Linguistics. doi: 10.18653/v1/2020.emnlp-main.305. URL https://aclanthology.org/2020.emnlp-main.305.

Timothée Lacroix, Guillaume Obozinski, and Nicolas Usunier. Tensor decompositions for temporal knowledge base completion, 2020.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*, 2019.

Costas Mavromatis, Prasanna Lakkur Subramanyam, Vassilis N. Ioannidis, Soji Adeshina, Phillip R. Howard, Tetiana Grinberg, Nagib Hakim, and George Karypis. Tempoqr: Temporal question reasoning over knowledge graphs, 2021.

Sumit Neelam, Udit Sharma, Hima Karanam, Shajith Ikbal, Pavan Kapanipathi, Ibrahim Abdelaziz, Nandana Mihindukulasooriya, Young-Suk Lee, Santosh Srivastava, Cezar Pendus, et al. A benchmark for generalizable and interpretable temporal question answering over knowledge bases. *arXiv preprint arXiv:2201.05793*, 2022.

Fabio Petroni, Tim Rocktäschel, Patrick Lewis, Anton Bakhtin, Yuxiang Wu, Alexander H Miller, and Sebastian Riedel. Language models as knowledge bases? *arXiv preprint arXiv:1909.01066*, 2019.

Nina Poerner, Ulli Waltinger, and Hinrich Schütze. E-bert: Efficient-yet-effective entity embeddings for bert. *arXiv preprint arXiv:1911.03681*, 2019.

Adam Roberts, Colin Raffel, and Noam Shazeer. How much knowledge can you pack into the parameters of a language model? *arXiv preprint arXiv:2002.08910*, 2020.

Paul Röttger and Janet B Pierrehumbert. Temporal adaptation of bert and performance on downstream document classification: Insights from social media. *arXiv preprint arXiv:2104.08116*, 2021.

Ahmad Sakor, Kuldeep Singh, Anery Patel, and Maria-Esther Vidal. Falcon 2.0: An entity and relation linking tool over wikidata. In *Proceedings of the 29th ACM International Conference on Information amp; Knowledge Management*, CIKM '20, pp. 3141–3148, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368599. doi: 10.1145/3340531.3412777. URL https://doi.org/10.1145/3340531.3412777.

Apoorv Saxena, Soumen Chakrabarti, and Partha Talukdar. Question answering over temporal knowledge graphs. *arXiv preprint arXiv:2106.01515*, 2021.

Petar Veličković, Guillem Cucurull, Arantxa Casanova, Adriana Romero, Pietro Lio, and Yoshua Bengio. Graph attention networks. *arXiv preprint arXiv:1710.10903*, 2017.

Manzil Zaheer, Guru Guruganesh, Kumar Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. Big bird: Transformers for longer sequences. *Advances in Neural Information Processing Systems*, 33:17283–17297, 2020.

Chen Zhu, Ankit Singh Rawat, Manzil Zaheer, Srinadh Bhojanapalli, Daliang Li, Felix Yu, and Sanjiv Kumar. Modifying memories in transformer models. *arXiv preprint arXiv:2012.00363*, 2020.