Benchmarking GPT-3 For Closed-Book QA: Strengths and Weaknesses

Chenglei Si[†], Naman Molri, Gurmehar Cheema, Elliot Huang, Arjun Akkiraju
CMSC 470[*]
University of Maryland, College Park

Abstract

Large-scale pretrained language models (PLMs) have achieved rapid progress in recent years, dominating the leaderboards of various NLP tasks. In this work, we focus on one such model, namely GPT-3 (Brown et al., 2020). We thoroughly evaluate GPT-3 on the task of closed-book question answering (CBQA). Question answering is a knowledge-intensive task requiring significant amount of factual knowledge. By constraining in the closed-book setting, models cannot retrieve knowledge from external knowledge bases but instead have to rely on the knowledge stored in their own parameters. By analyzing the performance of GPT-3 on different types of questions, we not only probe how much factual knowledge is stored in GPT-3, but also gain a better understanding of the strengths and weaknesses of GPT-3 in terms of different reasoning capabilities. To facilitate future work, we open source all the code, model predictions and human annotations in `https://github.com/NoviScl/GPT3QA`.

1 Introdcution

Large-scale PLMs such as BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), T5 (Raffel et al., 2020) and GPT-3 (Brown et al., 2020) have become the default choice in many NLP tasks due to their strong empirical performance. In particular, the large number of parameters in these PLMs give rise to the possibility of language models as knowledge bases (Petroni et al., 2019) where large amounts of factual knowledge is stored in the parameters of these PLMs, removing the need for external knowledge bases like Wikipedia.

This idea motivated a new approach of solving open-domain question answering (ODQA). Traditionally, ODQA adopts a retriever-reader pipeline (Chen et al., 2017) where a retriever first finds the most relevant passage from external knowledge bases and then runs a machine reading model on the selected passage to extract the answer span. In contrast, since PLMs already contain significant amount of knowledge, recent works (Brown et al., 2020) directly feed the questions to a generative model and let the model generate the answer. We call this setting **closed-book QA** (CBQA) because the model is not conditioning on passages from any external sources. In this work, we focus on benchmarking GPT-3 on a wide range of CBQA datasets as a way to understand the strengths and weaknesses of GPT-3.

Compared to other natural language understanding tasks, the evaluation of question answering models is a non-trivial task. Previous works (Si et al., 2021) have pointed out pitfalls in the conventional exact matching metric. Our preliminary human analysis also reveals that exact matching misses many correct predictions that have different surface forms than the gold answer annotation. In order to get reliable evaluation for our benchmarking, we conducted human annotation of model predictions and use them as references to guide the development of better automatic evaluation systems. We show that our new evaluation system is much better correlated with human judgement than the original exact matching metric. We release our evaluation system along with all model predictions and human annotations to facilitate future works on further improving the automatic evaluation of QA models.

Moreover, it is well known that prompt engineering plays an important part in the performance of GPT-3 (Lu et al., 2021; Liu et al., 2021). To better understand this impact in the context of CBQA, we further implement a model called **GPR**: GPT-3 with **P**rompt **R**etrieval. GPR dynamically retrieves

[†] Corresponding author: `clsi@umd.edu`
[*] A list of each author's contribution can be found at the end of the paper.

the most relevant QA-pairs from the training corpus for each test question and use them to construct the prompt. We benchmark the performance of both GPT-3 and GPR on our CBQA test sets.

In summary, our contributions in this paper are:

1. We manually annotate 1600 model predictions and analyze the mistakes of the automatic exact matching metric. Based on the analysis, we develop an improved **Fuzzy Match** system for evaluating model predictions. (Section 2)

2. We compile a comprehensive CBQA benchmark called **DiverseQA** consisting of different types of questions from eight existing QA datasets. (Section 3)

3. We implement three CBQA systems: a T5 based QA system, a GPT-3 based QA system and a prompt-retrieval augmented system built on top of GPT-3 called **GPR**. (Section 4)

4. We benchmark T5, GPT-3 and GPR on our DiverseQA benchmark and analyze the strengths and weaknesses of these models. (Section 5)

2 Automatic Evaluation: What's Wrong and How to Fix

2.1 Motivation

The most accurate way of evaluating model predictions is apparently through human annotation. However, human evaluation is expensive and time-consuming and so we have to resort to automatic metrics especially when our test sets are large. The most popular automatic evaluation metric in QA research is **Exact Match (EM)**, which does simple string matching between normalized answer prediction and gold answer. However, it is well-known that EM often misses correct predictions in alternative forms (Si et al., 2021).

As a motivating example, when we give the question "What is the current Mac OS operating system?" (from NQ), GPT-3 predicts "macOS Mojave" while the gold answer is "10.14". Non-surprisingly, EM considers this wrong, while all of authors of this paper think it is correct because "10.14" is the version number of "macOS Mojave".

In order to ensure accurate automatic evaluation of model predictions, we propose a new evaluation metric named **Fuzzy Match (FM)**. The development of FM is through iterative trial and error - the authors first manually annotated a set of model predictions and then try to resolve all possible errors of EM by incorporating corresponding matching heuristics. The following section details the FM system.

2.2 Fuzzy Match

First, we perform normalization for articles, punctuations, white spaces as well as Unicode to Ascii conversion on model predictions. RapidFuzz[1] is then used for fuzzy matching between strings. RapidFuzz computes a similarity score between two strings based on character matching and Levenshtein distance. Based on thorough tuning and human inspection, a similarity score of 63.3% is deemed enough to mark predictions correct. Another common problem in automatic evaluation is alternative surface forms or synonyms. For example, "snake" and "serpent" can often be considered as equivalent, but their string similarity is very low. To handle this, we mine NLTK's WordNet (a lexical database for synonyms and more) to add synonyms of gold answers into the answer set. The idea is similar to Si et al. but we use a synonym dictionary instead of Freebase. Put together, our FM evaluator computes the string similarity between the normalized prediction and all possible answers including synonyms in the answer set, and returns the highest matching score. To assess the reliability of our FM evaluator, we compare the FM scores with human evaluation scores, as well as the EM scores as baseline.

3 DiverseQA: The Test Suite for CBQA

In order to comprehensively analyze the strengths and weaknesses of the CBQA models, we assemble a test set named DiverseQA consisting questions from 8 different QA datasets. We briefly introduce each dataset below. For each dataset, we sample 1000 training questions and 100 test questions. Note that for all of them, we make no use of passages from sources like Wikipedia.

Natural Questions (NQ) (Kwiatkowski et al., 2019) consists of real search queries from Google search engine.

TriviaQA (Joshi et al., 2017) consists of trivia questions authored by trivia enthusiasts.

BeerQA (Qi et al., 2021) consists of multi-hop questions with varying reasoning steps.

[1] `https://github.com/maxbachmann/RapidFuzz`

Dataset	Evaluator	T5	GPT-3	GPR	ODQA
NQ	Exact Match	–	24	29	41.5 (DPR)
	Fuzzy Match	–	29	34	–
	Human Match	–	36	**38**	–
TriviaQA	Exact Match	25	53	52	56.8 (DPR)
	Fuzzy Match	36	61	62	–
	Human Match	–	59	**63**	–
BeerQA	Exact Match	–	26	27	51.1 (IRRR)
	Fuzzy Match	–	35	35	–
	Human Match	–	35	**36**	–
CSQA2	Exact Match	–	47	32	73.9 (T5-11B)
	Fuzzy Match	–	47	32	–
	Human Match	–	47	32	–
HotpotQA	Exact Match	–	26	23	60.4 (BeamDR)
	Fuzzy Match	–	34	32	–
	Human Match	–	34	30	–
QANTA	Exact Match	62	50	49	48.8 (BERT)
	Fuzzy Match	67	80	84	–
	Human Match	67	85	85	–
StrategyQA	Exact Match	–	79	73	72 (RoBERTa)
	Fuzzy Match	–	79	73	–
	Human Match	–	79	73	–
TimeQA	Exact Match	–	10	12	14.3 (Bigbird)
	Fuzzy Match	–	17	22	–
	Human Match	–	19	**24**	–

Table 1: Evaluation results of T5, GPT-3, GPR and strong open-domain QA models from recent papers. We evaluate 100 randomly sampled test questions for each dataset and report the score measured by different evaluators. We find that: 1) Our Fuzzy Match better aligns with human scores. 2) GPR is better than GPT-3 on 4 out of the 8 datasets, but notably does much worse on CSQA2 and StrategyQA. 3) GPT-3 based systems significantly outperform open-domain QA models on TriviaQA, QANTA, StrategyQA and TimeQA, but notably still lagging behind multi-hop questions and commonsense questions. Also, it is worth noting that all current models perform poorly on time-sensitive questions, showing large room for future work. Last but not least, please note that since StrategyQA's test set is hidden, our results are evaluated on a hold-out set from the training set. When we submit the system to the official leaderboard (`https://leaderboard.allenai.org/strategyqa/submissions/public`), the score is 59.18 on the official test set, indicating potential distribution shifts between the training and test set.

CSQA2 (Talmor et al., 2021) consists of yes/no questions based on commonsense knowledge.

HotpotQA (Yang et al., 2018) consists of multi-hop questions where the questions are based on Wikipedia.

QANTA (Rodriguez et al., 2019) consists incremental questions in the form of Quizbowl. Typically models need to buzz in once they are confident about the answer. In this work we feed the entire question prompt to the model without involving the buzzer for consistency with other QA formats.

StrategyQA (Geva et al., 2021) consists of multi-hop questions that require impilict reasoning. For example, "Did Aristotle use a laptop?".

TimeQA (Chen et al., 2021) consists of time-sensitive questions. For example, " What position did George Washington hold in 1777?", and " What position did George Washington hold from 1789 to 1797?".

4 Models: T5, GPT-3 and GPR

In this work we have explored three types of models for CBQA: T5, GPT-3, GPR.

4.1 T5

We used Text-to-Text Transfer Transformer (T5) model to essentially act as a baseline. Our T5 model was trained on a mixture of TriviaQA and QANTA datasets. T5 model's ability to use input text as context to generate output text made it a great candidate for CBQA trivia-style questions where we train the model on question/answer pairs without additional context.

The first step towards training the T5 model for CBQA was to normalize and standardize the data from QANTA and TriviaQA datasets. QANTA dataset was more challenging to normalize and standardize as most of the questions contained lot of information usually at the beginning that was contextual knowledge for the answer, but the answer itself was obviously not contained in there due to QANTA questions being in trivia format. The key was to find the beginning of the question which usually started with patterns like ”For 10 points”, ”for ten points”, ”ftp”, ”for fifteen points” etc. For the answer associated with a given question, we extracted ”page” field from the QANTA dataset as it indicated the name of the Wikipedia page that contained the answer. After we normalize the question and answer text (for e.g. making them case insensitive, removing symbols, punctuations etc.) we prepended ”trivia question: ” in front of the formatted QANTA and TriviaQA dataset question as part of the standardizing process. After both datasets were preprocessed, we created a dataset mixture using MixtureRegistry from SeqIO, which is a library used by T5 for managing data pipelines and for evaluation metrics. This dataset mixing was done in order to avoid overfitting on questions in a given dataset. Finally, we trained the T5-large model which is based on Mesh Tensorflow model (Shazeer et al., 2018) on Google Cloud TPU. We decided to use upto 256 embeddings for our question text even though a given layer in T5 model is insensitive to relative position beyond 128 tokens but the subsequent layers can build a sensitivity to larger offsets by combining local information from previous layers. (Raffel et al., 2020).

4.2 GPT-3

GPT-3 is a transformer-based autoregressive language model pretrained on massive amount of text data from the web. The training objective is the language modeling objective where the model predicts the next word given the context. We perform few-shot learning on GPT-3 via in-context prompting. Specifically, we randomly sample k QA pairs from the training set and concatenate them as the prompt to the model. We format the QA pairs as: “[question]. The answer is [answer].” in the prompt. When predicting answers for the test questions, we just let the model complete the “[answer]” part. For hyper-parameters, we set the number of examples k to be 16 for all datasets except QANTA where we use k=4 because QANTA questions are much longer (average question length of the test set is 134). We use temperature 0 to reduce randomness in the decoding and set the max number of decoded tokens to 6.

4.3 GPR: GPT-3 with Prompt Retrieval

Randomly selecting demonstration examples from the training set for the prompt unavoidably introduces randomness to the results: bad demo examples could lead to poor performance. To avoid this, we retrieve the k most similar QA pairs from the training set to use as the prompt. Specifically, we use TF-IDF to encode all questions and retrieve the most similar questions for each given test question based on cosine similarity. As an example, we show a test question and the top 4 retrieved questions from TriviaQA.

- Test Question: ‘Which President of the Philippines was deposed in 1986?’
- Top 1: ‘In which year was Ferdinand Marcos first elected as President of the Philippines?’
- Top 2: ’Who became president of the USA in 1989?’
- Top 3: ’Who was the first Democrat President of the 20th century?’
- Top 4: ’Who was president of the USA at the outbreak of World War I?’

5 Benchmarking: Results and Insights

We present the evaluation results in Table 1. We highlight and discuss a few interesting observations below:

- Exact Match poorly captures the true performance of models. In comparison, our Fuzzy Match system almost perfectly correlates with human evaluation results.
- T5 is much worse than GPT-3 based systems even if T5 is trained on abundant training data while GPT-3 is used in a few-shot manner.

- In some instances, T5 produced a more complete answer than the label. For example here is a question from TriviaQA: ”Which island is named after the world’s largest bear?” The answer associated with the question(not including the aliases) was ”Kodiak”. T5 model’s prediction was ”Kodiak Island”.

- GPR outperforms GPT-3 in 4 out of the 8 datasets, while it falls short on StrategyQA, CSQA2 and HotpotQA. One possible explanation is that relevant prompts are not helpful for commonsense reasoning and multihop reasoning.

- Even in a few-shot manner, GPT-3 and GPR can outperform ODQA models trained on full training sets on TriviaQA, QANTA, StrategyQA and TimeQA. While TriviaQA and QANTA focus more on factoid questions in nature, StrategyQA tests implicit multi-hop reasoning. It may be surprising that GPT-3 can achieve such high performance on StrategyQA without any explicit knowledge of the evidence chain. On the other hand, we have a rather confident conclusion that GPT-3 is still not good at time-sensitive questions yet, which motivates works on temporal aware language models.

6 Conclusion

In this work, we benchmarked GPT-3 on different types of questions to understand its strengths and weaknesses. We find that it performs well on factoid questions but is still weak on multi-hop, commonsense and time-sensitive questions. As an additional contribution, we developed a more reliable evaluation system for evaluating models predictions, which has been shown to better reflect models’ true performance. Due to time constraints, we leave more in-depth analysis to future work and hope that our results not only inform the community of GPT-3’s capabilities, but also shed light on what aspects of question answering needs more improvement.

Contributions

Chenglei proposed the goal for the project, implemented and experimented GPT-3 and GPR systems, and drafted most sections of the paper.
Gurmehar experimented the T5 model, and wrote section 4.1 (the T5 section) of the paper.
Naman led the development of FuzzyMatch, annotated the model predictions, and wrote part of section 2.2 on Fuzzy Match.
Elliot proposed to incorporate WordNet and improved FuzzyMatch, and annotated model predictions.
Arjun did some orthogonal background research and contributed to the initial discussion.

We also thank Chen Zhao, Neha Srikanth and Jordan Boyd-Graber for their helpful discussion and feedback.

References

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

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. In *ACL*.

Wenhu Chen, Xinyi Wang, and William Yang Wang. 2021. A dataset for answering time-sensitive questions. *ArXiv*, abs/2108.06314.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. Bert: Pre-training of deep bidirectional transformers for language understanding. *ArXiv*, abs/1810.04805.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did aristotle use a laptop? a question answering benchmark with implicit reasoning strategies. *Transactions of the Association for Computational Linguistics*, 9:346–361.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. In *ACL*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc V. Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Jiachang Liu, Dinghan Shen, Yizhe Zhang, Bill Dolan, Lawrence Carin, and Weizhu Chen. 2021. What makes good in-context examples for gpt-3? *ArXiv*, abs/2101.06804.

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

Yao Lu, Max Bartolo, Alastair Moore, Sebastian Riedel, and Pontus Stenetorp. 2021. Fantastically ordered prompts and where to find them: Overcoming few-shot prompt order sensitivity. *ArXiv*, abs/2104.08786.

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

Peng Qi, Haejun Lee, OghenetegiriTGSido, and Christopher D. Manning. 2021. Answering open-domain questions of varying reasoning steps from text. In *EMNLP*.

Colin Raffel, Noam M. Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *ArXiv*, abs/1910.10683.

Pedro Rodriguez, Shi Feng, Mohit Iyyer, He He, and Jordan L. Boyd-Graber. 2019. Quizbowl: The case for incremental question answering. *ArXiv*, abs/1904.04792.

Noam Shazeer, Youlong Cheng, Niki Parmar, Dustin Tran, Ashish Vaswani, Penporn Koanantakool, Peter Hawkins, HyoukJoong Lee, Mingsheng Hong, Cliff Young, Ryan Sepassi, and Blake Hechtman. 2018. Mesh-TensorFlow: Deep learning for supercomputers. In *Neural Information Processing Systems*.

Chenglei Si, Chen Zhao, and Jordan L. Boyd-Graber. 2021. What's in a name? answer equivalence for open-domain question answering. In *EMNLP*.

Alon Talmor, Ori Yoran, Ronan Le Bras, Chandrasekhar Bhagavatula, Yoav Goldberg, Yejin Choi, and Jonathan Berant. 2021. Commonsenseqa 2.0: Exposing the limits of ai through gamification.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*.