Fact Retrieval and Ensemble Models on OpenBookQA

Andrew Nakamoto
447
anak2004@uw.edu

Chung Yik Edward Yeung
447
chungy04@uw.edu

Project Information:

Project Type	Default 447 Project
Task Name	OpenBookQA
Benchmark Link	URL to OpenBookQA

Individual contributions.

- **Andrew Nakamoto** contributed most towards the research for model architecture, the code for data preprocessing, the code for training and evaluating models, and the code, training, and evaluation of the fact-retrieval models. He wrote the Introduction, Background, Method's Fact-retrieval section, Experiments, Results, Discussion, and Conclusion.
- **Chung Yik Edward Yeung** contributed most towards evaluating and training the pre-trained and fine-tuned models, the code, training, and evaluation of the ensemble models, and the integration and evaluation of the GPT-4 model. He wrote the Method's Ensemble section, Experiments, Results, and Discussion.

1 Introduction

We are investigating the OpenBookQA task benchmark originally proposed by the paper Can a Suit of Armor Conduct Electricity? A New Dataset for Open Book Question Answering (Mihaylov et al. (2018)). The dataset consists of 5,957 multiple-choice elementary-level science questions (4,957 train, 500 dev, 500 test), which probe the understanding of a small "book" of 1,326 core science facts and the application of these facts to novel situations.

We chose this task because the combination of facts with multiple choice questions encourages models to both use latent knowledge as well as search for specific pieces of helpful information. This made it a very interesting task to research because there are essentially two different problems to tackle within the single benchmark. This task is important because it evaluates critical common-sense and fact-based reasoning skills that language models have historically struggled with. If this problem can be solved well, it can potentially lead to models that produce false or erroneous text at much lower frequency. Ideally, models with this trait will be safer as they cannot recommend untrue solutions and remedies to users or perpetuate harmful falsehoods.

We benchmarked a variety of pre-trained and fine-tuned models on OpenBookQA. We found that these performed poorly on the benchmark, suggesting that specialized models are necessary. We then built and tested two distinct strategies on the benchmark that were shown to be effective on the leaderboard: fact-retrieval models and ensemble models. We found that fact-retrieval was incredibly successful, boosting accuracy on models of all sizes by 2 to 4 percent. Our implementation of ensemble was unsuccessful, lowering accuracy across the board by around 2 percent.

We contribute several baselines for a wide variety of mid-sized off-the-shelf models and one large model. We demonstrate the efficacy of fact-retrieval methods for fact-based reasoning, and propose several further enhancements to be explored. We additionally provide a survey of the current top methods.

2 Background

We've summarized and indexed a survey of submissions to the benchmark.

OpenBookQA Leaderboard Survey			
Name	**Method**	**Acc.**	**Paper**
GPT-4 + KB	Sentence-BERT to retrieve the most related 3 facts, then concatenate the facts to the question and call the GPT-4 API.	0.959	Not provided
MVP-Tuning Ensemble	Leverages similar question-answer pairs in the training set to improve knowledge retrieval and employs a single prompt-tuned PLM to model knowledge and input text jointly.	0.952	Huang et al. (2023)
X-Reasoner	A retriever based on RocketQA and SentenceBERT is used to retrieve 10 science facts. A reader module based on T5-11B is used to encode questions, candidates, and facts and give the final answer.	0.942	Not provided
GenMC (ensemble)	Model first generates a clue from the prompt. This clue is then used to assist in answering the multiple-choice question. The ensemble model trains 10 models and has them vote on the result.	0.920	Huang et al. (2022)
GenMC	Model first generates a clue from the prompt. This clue is then used to assist in answering the multiple-choice question.	0.898	Huang et al. (2022)

Continued on next page

Table 1 – continued from previous page

Name	Method	Acc.	Paper
DeBERTa + UFO	Disregards the in-domain OpenBook corpus, instead using Universal Fact Obtaining strategy specified in their paper to train the model for commonsense reasoning.	0.896	Li et al. (2023)
DRAGON	Uses a bidirectional knowledge graph approach with novel pretraining. The model takes text segments as well as KG subgraphs as input and uses both for output.	0.878	Yasunaga et al. (2022)
GSC + AristoRoBERTa	Uses Graph Soft Counter module to process KGs, showing that a very simple graph neural counter can outperform other models on benchmarks.	0.874	Wang et al. (2021)
UnifiedQA (T5-11B; finetuned) - with IR	Argues specialized models are unnecessary. Uses a text-to-text model trained on four separate question formats from 20 existing datasets.	0.872	Khashabi et al. (2020)

While there are a variety of different published approaches to the benchmark, they fall broadly into two categories. The first category relies on knowledge graphs and variations thereof, shown by models like DRAGON and GSC. The second category relies on enhancing the prompt by retrieving or generating additional text knowledge, like with GPT-4+KB, MVP-Tuning Ensemble, and even GenMC. Each of these models divides the retrieval and answering sections into two distinct parts, and each of them accomplishes them in different ways. We tested fact-retrieval methods on a variety of models, and fully replicated the setup of GPT-4 + KB.

In addition, we see that both GenMC and MVP-Tuning find an accuracy boost of a few percent by using ensemble models. We tested whether this boost was inherent to their methods, or could be applied to any generic model setup to provide improvement on OpenBookQA.

3 Method

3.1 Fact Retrieval

The first method is a two-part model. The first part of the model takes the prompt and retrieves k (we tested 3 and 10) related science facts from the benchmark's provided corpus of science facts. This result is then concatenated to the prompt and the second part of the model selects an answer from the multiple choice options. This method is particularly advantageous for us because by using two pretrained models in combination, we eliminate the need for the extensive compute required by a brand new model.

The intuition behind this model is that the latent knowledge in large pretrained models can be brought out through in-context learning. By providing related science facts, we hope to boost model performance on the benchmark by improving its common sense and scientific reasoning through this mixture of fine tuning on the dataset and the generation of in-context facts to assist conclusions. It also makes sense that a model could be stronger if it is split into two distinct portions which can individually perform their specific tasks better than any model forced to do both at once. By having one model focus on finding relevant facts, and another on turning facts into answers, we can specialize each model to hopefully improve performance.

Our final model used Sentence-BERT for semantic similarity using cosine distance as the fact-retriever, where each fact in the corpus was compared against the question prompt for semantic similarity. The closest k facts were then selected and added to the beginning of the prompt. We then fine-tuned bert-based-uncased and DeBERTa-v3-large on the augmented prompts. We experimented with other fact retrieval methods, but Sentence-BERT was

supported by several papers on the leaderboard and additionally was already fine-tuned for us, so it ended up being both the least-compute-intensive and most effective option.

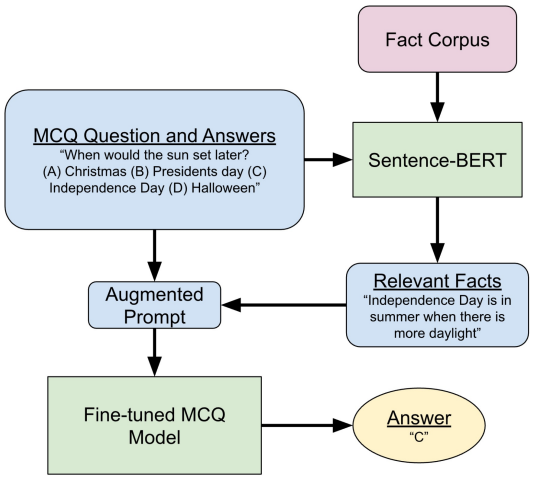

Figure 1: K-Top Facts Retrieval Pipeline

3.2 Ensemble

Our ensemble model pipeline integrates multiple fine-tuned models for answering multiple-choice questions (MCQs).

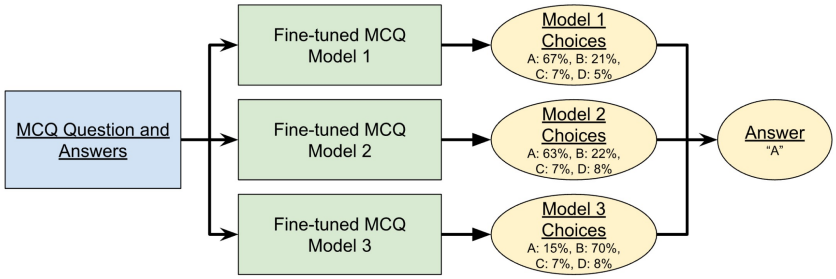

Figure 2: Ensemble Pipeline

Given an MCQ with its answer options, each independently trained fine-tuned model processes the question and predicts the likelihood of each option being the correct one, producing a set of probabilities for the choices. The final answer is then determined through an aggregation of these probabilities across the models, with the answer choice having the highest cumulative probability being selected as the output of the pipeline.

This approach is chosen to introduce stochastic variation in the training process, as each model instance may converge to slightly different solutions due to the randomness inherent in model initialization and mini-batch gradient descent. This variance allows each model to capture unique decision boundaries, which can be advantageous when predicting on complex MCQs. By aggregating the predictions, the ensemble harnesses these diverse perspectives, reducing the impact of overfitting and improving generalization on unseen data. We chose to use 3 models as a smaller ensemble might not offer the same breadth of perspectives, while a larger one could lead to unfeasible increased computational demands.

4 Experiments

4.1 Pre-trained Model Baselines

We evaluated the accuracy of the pre-trained models in their original, unmodified state on the validation set to establish a baseline performance. This additional work was useful for several reasons. First, it allowed us to measure the out-of-the-box capabilities of each model for answering multiple-choice questions. This provided insight into how well each model's pre-training generalized to our domain of interest and is a valuable benchmark for other models. Second, establishing a baseline was necessary to quantify the benefits of subsequent fine-tuning. By comparing the performance of the models before and after fine-tuning, we could directly assess the impact of fine-tuning on each model. Last, these baseline metrics served as an initial screening to identify which models held the most promise for further experimentation. Models that performed reasonably well without fine-tuning were prime candidates for enhanced performance through fine-tuning and fact-retrieval enhancements.

We measured baselines on the following pre-trained models: bert-base-uncased, bert-large-uncased, roberta-base, roberta-large, albert-base-v2, distilbert-base-uncased, xlnet-base-cased, google/electra-small-discriminator, t5-large, and google/bigbird-roberta-base.

4.2 Fine-tuned Model Baselines

After establishing baseline performances, we opted to fine-tune a curated selection of models based on their initial performance and distinctive capabilities. The models chosen include bert-base-uncased, roberta-large, albert-base-v2, distilbert-base-uncased, and google/electra-small-discriminator. Each model brings a unique strength to the table: BERT base model (uncased) and DistilBERT base model (uncased) are adept at grasping context due to their training on a large corpus of English data using a masked language modeling objective, where they learn a bidirectional representation of sentences (Sanh et al. (2019))(Devlin et al. (2018)). RoBERTa large, which is case-sensitive and trained similarly on a vast English dataset, excels in processing complex texts (Liu et al. (2019)). ALBERT Base v2 stands out for its efficiency and smaller memory footprint, achieved by layer-sharing in its architecture and a focus on both masked language modeling and sentence ordering prediction during its pertaining (Lan et al. (2019)). Lastly, the google/electra-small-discriminator is remarkable for its capability to differentiate between correct and incorrect predictions, employing a unique training approach where it is taught to distinguish "real" input tokens from "fake" ones generated by another network (Goodfellow et al. (2014)).

We fine-tuned each model for 5 epochs with a learning rate of 5e-05, batch size of 32, Adam optimizer with betas=(0.9, 0.999) and epsilon=1e-08, and linear lr_scheduler_type. We saved the epoch with the highest accuracy on the validation set. We found that in all cases, accuracy plateaued after a single epoch and so we did not experiment with longer training due to worries about overfitting to the relatively small training dataset.

For our fact-retrieval and ensemble methods, we chose to use bert-base-uncased, DeBERTa-v3-large, and GPT-4. The bert-base-uncased model was chosen as a mid-sized model that performed well on the benchmark. DeBERTa-v3-large was chosen as a large model that had shown success on the leaderboard. GPT-4 was chosen for its top-of-the-leaderboard performance, despite its API abstraction. Therefore we also benchmarked DeBERTa-v3-large and GPT-4.

GPT-4 accuracy was evaluated without fine tuning (as it is not available). DeBERTa-v3-large was fine-tuned two separate times for 7 epochs with learning rates of 3e-05 and 9e-06, a batch size of 16 (had to be reduced to fit in memory), Adam optimizer with betas=(0.9, 0.999) and epsilon=1e-08, and linear lr_scheduler_type. As the model with learning rate 3e-5 had the best results, we used it going forward. Then the models were evaluated for accuracy on the validation set.

4.3 Fact-retrieval

As the Sentence-BERT weights did not need to be updated, the augmented prompt could be produced in the preprocessing step when training to save time. bert-base-uncased and DeBERTa-v3-large were both fine tuned for k-fact-retrieval with $k = 3$ and $k = 10$ on these augmented prompts to create a fact-retrieval model.

For $k = 3$, bert-base-uncased was fine tuned for 5 epochs with learning rate 5e-05, a batch size of 32, Adam optimizer with betas=(0.9, 0.999) and epsilon=1e-08, and linear lr_scheduler_type. For $k = 10$, bert-base-uncased was fine tuned in the same way except the batch size was reduced to 16.

For $k = 3$, DeBERTa-v3-large was fine tuned for 5 epochs with learning rate 3e-05, a batch size of 16, Adam optimizer with betas=(0.9, 0.999) and epsilon=1e-08, and linear lr_scheduler_type. For $k = 10$, bert-base-uncased was fine tuned in the same way except the batch size was reduced to 8.

These batch size changes were necessary to reduce memory usage when training. All models had their accuracy evaluated on the fact-augmented validation set, and the best epoch was saved.

For GPT-4, we appended the top k most relevant facts chosen by Sentence-BERT to the beginning of the prompt before feeding the question into the API. We did this for both $k = 3$ and $k = 10$, and evaluated accuracy on the validation set.

4.4 Ensemble

To test whether the GenMC and MVP-Tuning's ensemble accuracy improvements on OpenBookQA applied to other methods as well, we evaluated ensemble accuracy for fine tuned models. For both bert-base-uncased and DeBERTa-v3-large, a model was fine-tuned 3 separate times on the training set. Due to the random initialization of the weights and the shuffled training data, this produced slight variations across the 3 models. These 3 models were then used in an ensemble setup when evaluated, where their MCQ output probabilities were averaged to produce a single answer.

Both ensemble bert-base-uncased and DeBERTa-v3-large were fine tuned for 5 epochs with learning rate 3e-05, a batch size of 16, Adam optimizer with betas=(0.9, 0.999) and epsilon=1e-08, and linear lr_scheduler_type.

For the ensemble using GPT-4, prompts were sent to the GPT-4 API three separate times. Each response was recorded, the most common response was considered the ensemble answer. In the event of a tie, the answer was chosen randomly from each individual answer.

All three of these setups were evaluated for accuracy on the validation set.

4.5 Datasets

We trained on the training set included in the OpenBookQA benchmark dataset. We evaluated on the validation set included in the OpenBookQA benchmark dataset. The training dataset contains 4957 questions, and the validation dataset contains 500 questions.

4.6 Baselines

We compared our fine tuned models against raw pretrained models on the OpenBookQA dataset. See Figure 3 below for data on this baseline. We discuss this further in Discussion.

Additionally, we collated an index of the top performing models on the OpenBookQA leaderboard in Background above, which are compared against out final models.

4.7 Code

Relevant code:

- Baseline data preprocessing and training
- Fact-retrieval data preprocessing and training
- GPT-4 fact-retrieval
- bert-based-uncased and DeBERTa-v3-large ensemble
- GPT-4 ensemble
- Folder with remaining miscellaneous test, visualization, and duplicate code

Relevant models can be found on HuggingFace under the user profiles of asn1814 and edwardyeung04.

5 Results

5.1 Pre-trained and fine-tuned baselines

The following pretrained models were evaluated for accuracy on the validation set without fine tuning:

Model / Tokenizer	Acc
bert-base-uncased	0.242
bert-large-uncased	0.282
roberta-base	0.24
roberta-large	0.278
albert-base-v2	0.256
distilbert-base-uncased	0.308
xlnet-base-cased	0.22
google/electra-small-discriminator	0.246
t5-large	0.272
google/bigbird-roberta-base	0.256

Table 2: Accuracy of Different Models Before Finetuning

The following models were fine tuned on the training set, followed by evaluation on the validation set:

Model / Tokenizer	Best Epoch	Best Accuracy
bert-base-uncased	4	0.572
distilbert-base-uncased	2	0.558
google/electra-small-discriminator	4	0.540
roberta-large	3	0.276
albert-base-v2	2	0.558

Table 3: Accuracy of Different Models Following Finetuning

These combined results are summarized in Figure 3. Overall, these results show that pre-trained models perform abysmally on OpenBookQA, performing little better than random guessing. This implies that this task is far more difficult that basic natural language generation. After fine-tuning, we see a boost that consistently brings mid-size models to around 55% accuracy - still far below top leaderboard performance. This confirms the claim that strong neural baselines achieve around 50% accuracy on the dataset.

5.2 Fact-retrieval and Ensemble

The validation accuracies of our various models are shown in Figure 4. We compare bert-base-uncased, DeBERTa-v3-large, and GPT-4. Each model was evaluated under different conditions: with no fine-tuning, with fine-tuning, with fact retrieval at $k = 3$, with fact retrieval at $k = 10$, and as part of an ensemble approach. For the bert-base-uncased models, fine-tuning led to an improvement of approximately 30% over the baseline accuracy.

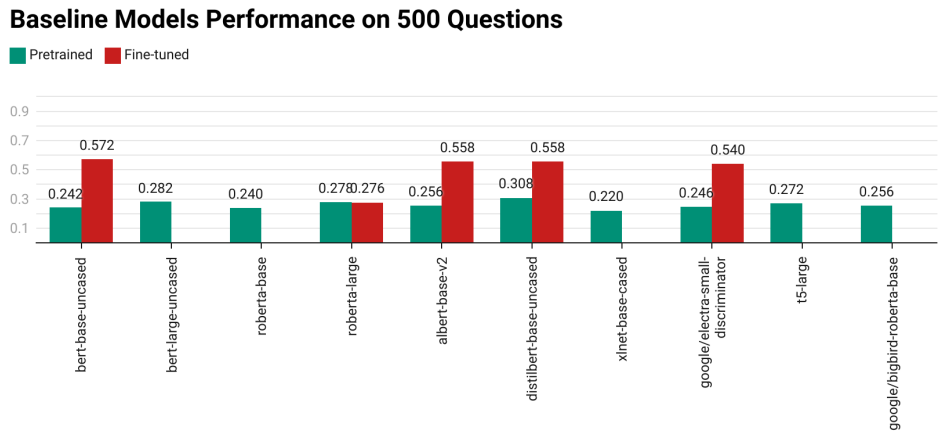

Figure 3: Baseline on OpenBookQA for various pretrained and fine tuned models

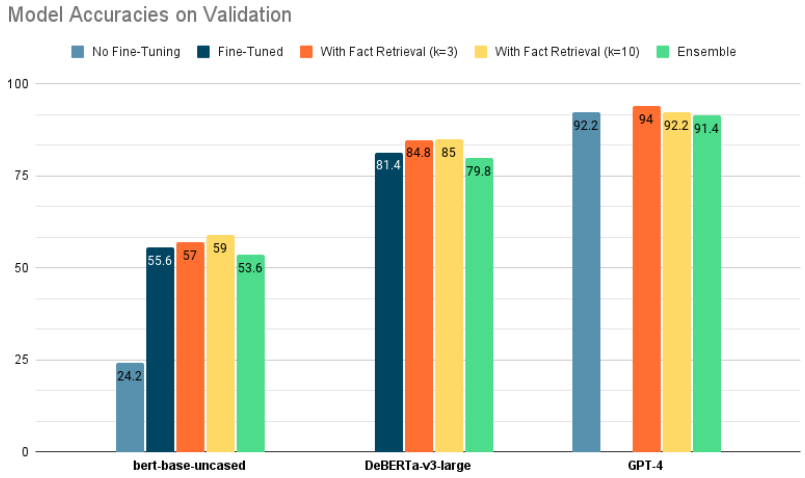

Figure 4: Accuracies on Validation Set for Multiple Models

Incorporating fact retrieval further improved the performance by several percent, with $k = 10$ outperforming $k = 3$.

The DeBERTa-v3-large models performed even better, which was expected as the model is much larger. They similarly saw improvement from our fact retrieval method, although the boost from increasing the number of facts retrieved was less significant.

The GPT-4 models achieved the highest accuracies across all individual setups. Notably, fact retrieval with $k = 10$ did not result in improvement, whereas $k = 3$ did. This suggests that excessive prompt length and irrelevant facts can undo the benefits of fact-retrieval for GPT-4.

The ensemble method consistently showed a decrease in performance of about 2% from the fine-tuned models. This suggests that the benefits GenMC and MVP-Tuning found from ensemble do not apply to the dataset itself, but instead are specific to their model setups.

5.3 OpenBookQA Leaderboard Comparison

In comparison to the OpenBookQA leaderboard standings, our top-performing model (GPT-4 with Fact Retrieval at k=3) would rank 4th. Other variations of GPT-4 also secured positions within the top 10, demonstrating robust performance across different configurations. Our DeBERTa-v3-large $k = 10$ fact retrieval model aligns with the 21st spot on the leaderboard, with our other DeBERTa variations ranking between the 20th and 30th positions. Meanwhile, the bert-base-uncased models ranked towards the lower end of the leaderboard, indicating a relative underperformance compared to the other models we tested (likely due to their small size).

6 Discussion

6.1 Additional difficulties of OpenBookQA

The wide variety of question formats in the benchmark is an adverse factor beyond the difficulty of the questions themselves that may impact performance. While it's true that all the questions are multiple choice, the structure of their presentation varies greatly.

Consider these three question and answer pairs: "Tunnels (A) lead to less impacted soil (B) pact down soil to make it denser (C) firm up the ground (D) help prevent the effects of erosion", "What is a source of energy? (A) bricks (B) grease (C) cars (D) dirt", and "Why can an Owl retain it's body temperature during the winter? (A) It's neither (B) It's both (C) It's cold blooded (D) It's warm blooded". The first is essentially sentence completion or token prediction. The second is a standard multiple choice question, phrased as a question. The third is the same, but includes the answers "it's both" and "it's neither", potentially confusing a model that only looks at one prompt-answer pair at a time.

This may increase the difficulty faced when fine tuning a model, because it will have fewer examples of each type of question format and will not be able to specialize on any one format, and unfairly harm analysis that assumes the benchmark is purely focused on scientific reasoning ability.

6.2 Training over several epochs

Model validation accuracy is essentially maximized after just one pass over the training data, and varies a few percentage points noisily afterward. This is likely because each science fact is so specific and the dataset is so small (only 5,000 facts), so after just one pass overfitting starts to occur. For this reason, we aimed to train the models for very few epochs to prevent this issue.

One idea for future research is that if the scientific knowledge itself is not being trained on, but instead the format of the multiple-choice questions is, one could potentially find an unrelated MCQ dataset to increase the size of the training corpus and improve a model's ability to process multiple choice questions despite the new dataset not involving scientific-reasoning-related questions.

6.3 Fact-Prompt similarity metrics

To determine the facts most relevant to a given question, we used Sentence-BERT to compute an embedding of all 5,000 facts in the corpus along with an embedding of the question, and then used distance measured by cosine similarity to find the nearest facts to the prompt.

However, very recent research raises the possibility that cosine similarity between embeddings can produce arbitrary results (Steck et al. (2024)). While this study specifically focuses on regularized linear models, the idea is not contained to that paradigm. Empirically, cosine similarity worked well for us, and there are countless papers that report the successful use of cosine similarity in practical applications. However, potential future work could explore the impact of different distance measures, such as dot product similarity or training directly

for cosine similarity. With sufficient time and compute, a fact-retrieval model could be trained or fine-tuned specifically for the task of science facts.

6.4 Ensemble performance

Our study's ensemble approach proved not only ineffective but actively harmful. Utilizing an ensemble composed of three instances of the same model may not have sufficiently diversified the decision-making process, potentially resulting in correlated errors and limited variance reduction. As we only tested ensemble methods on simple fine-tuned model architectures, an obvious next step would be to test our fact-retrieval models in ensemble.

We aggregated model predictions by summing probabilities, aiming to capture a more detailed consensus among models, especially in closely contested decisions. This method should theoretically offer a nuanced integration of model outputs, yet the ensemble's performance suggests that simply combining probabilities may not effectively capitalize on the potential of collective model intelligence. Our ensemble approach's lackluster performance suggests that summing probabilities might not have been the most effective strategy for our homogeneous model setup. Employing argmax across all model outputs is a promising alternative, potentially offering clearer, more decisive selections by focusing on the answer with the highest overall confidence. Another option is integrating a diverse set of models into the ensemble. This could potentially exploit the unique capabilities of each. On the other hand, if one model consistently outperforms the rest, its inclusion with lower-performing models actually compromise the ensemble's overall effectiveness.

Future investigations might benefit from exploring more adaptive ensemble configurations that leverage the complementary strengths of a broader array of models, aiming for a synergy that enhances performance without prohibitive increases in computational load. Overall, it may simply be that ensemble is only useful for certain architectures and the gains seen by GenMC and MVP-Tuning's use of ensemble are idiosyncratic to their setups.

7 Conclusion

We present benchmarks for several different models on OpenBookQA, and additionally implement and evaluate two separate leaderboard-proven methods. We show that fact-retrieval is simple and effective, and present initial trends that the number of facts should scale inversely with model size. We show that ensemble methods do not universally improve performance on OpenBookQA but instead are limited to specific model architectures.

Fact-retrieval proves to be a very effective method for the benchmark. The models we implemented that use it show significant gain over their baseline counterparts, and rank high on the leaderboard. Our GPT-4 model would in-theory place 4th (although I expect more because of GPT-4, despite our accuracy boost), and our fine-tuned DeBERTa-v3-large 10-fact-retrieval model would be 21st.

Our work offers several potential future explorations. The first is a more intense study of how model size impacts the optimal number of facts to retrieve. Another is testing different semantic similarity metrics for fact retrieval. We also show that ensemble techniques fail with a small number of basic models, but we did not test larger numbers of models or more complex model setups. It is reasonable that ensemble strategies would provide larger improvements in these situations where variance is higher.

Overall, we gain insight into two distinct strategies for common-sense reasoning and present models that perform impressively on OpenBookQA.

References

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Ian J. Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial networks, 2014.

Yongfeng Huang, Yanyang Li, Yichong Xu, Lin Zhang, Ruyi Gan, Jiaxing Zhang, and Liwei Wang. MVP-tuning: Multi-view knowledge retrieval with prompt tuning for commonsense reasoning. In Anna Rogers, Jordan Boyd-Graber, and Naoaki Okazaki (eds.), *Proceedings of the 61st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pp. 13417–13432, Toronto, Canada, July 2023. Association for Computational Linguistics. doi: 10.18653/v1/2023.acl-long.750. URL https://aclanthology.org/2023.acl-long.750.

Zixian Huang, Ao Wu, Jiaying Zhou, Yu Gu, Yue Zhao, and Gong Cheng. Clues before answers: Generation-enhanced multiple-choice qa, 2022.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. Unifiedqa: Crossing format boundaries with a single qa system, 2020.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. ALBERT: A lite BERT for self-supervised learning of language representations. *CoRR*, abs/1909.11942, 2019. URL http://arxiv.org/abs/1909.11942.

Zhifeng Li, Yifan Fan, Bowei Zou, and Yu Hong. Ufo: Unified fact obtaining for commonsense question answering, 2023.

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. *CoRR*, abs/1907.11692, 2019. URL http://arxiv.org/abs/1907.11692.

Todor Mihaylov, Peter Clark, Tushar Khot, and Ashish Sabharwal. Can a suit of armor conduct electricity? a new dataset for open book question answering. In *Conference on Empirical Methods in Natural Language Processing*, 2018. URL https://api.semanticscholar.org/CorpusID:52183757.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *ArXiv*, abs/1910.01108, 2019.

Harald Steck, Chaitanya Ekanadham, and Nathan Kallus. Is cosine-similarity of embeddings really about similarity?, 2024.

Kuan Wang, Yuyu Zhang, Diyi Yang, Le Song, and Tao Qin. Gnn is a counter? revisiting gnn for question answering, 2021.

Michihiro Yasunaga, Antoine Bosselut, Hongyu Ren, Xikun Zhang, Christopher D Manning, Percy Liang, and Jure Leskovec. Deep bidirectional language-knowledge graph pretraining, 2022.

A Appendix

Acknowledgments

This template is modified from the COLM 2024 paper template. Instructions are written by Liwei Jiang, Alisa Liu, and Yegor Kuznetsov.