# Can Open-Domain QA Reader Utilize External Knowledge Efficiently like Humans?

**Neeraj Varshney, Man Luo, Chitta Baral**

Arizona State University
{nvarshn2, mluo26, chitta}@asu.edu

## Abstract

State-of-the-art open-domain QA models typically use a retriever–reader approach in which the retriever finds the relevant knowledge/passages and the reader leverages that to predict the answer. Prior work has shown that the performance of the reader usually tends to improve with the increase in the number of these passages. Thus, state-of-the-art models use a large number of passages (e.g. 100) for inference. While the reader in this approach achieves high prediction performance, its inference is computationally very expensive. We humans, on the other hand, use a more efficient strategy while answering: firstly, if we can answer the question using our already acquired knowledge then we do not even use the external knowledge, and in the case when we do require external knowledge, we don't always need to read the entire knowledge; we only read the amount of knowledge that is sufficient to find the answer. Motivated by this procedure, we ask a research question "*Can the open-domain QA reader utilize external knowledge efficiently like humans without sacrificing the prediction performance?*"

To this end, we explore an approach that utilizes both the 'closed-book' (knowledge present in the model parameters) and the 'open-book' (external knowledge) inference. Furthermore, instead of using a large fixed number of passages for open-book inference, we dynamically read the external knowledge in multiple 'knowledge iterations'. Through comprehensive experiments on NQ and TriviaQA datasets, we demonstrate that this dynamic reading approach improves both the **inference efficiency** and the **prediction accuracy** of the reader. Comparing with the Fusion-in-Decoder reader, we show that this approach matches its accuracy by utilizing just $18.32\%$ of its reader inference cost (measured in FLOPs) and also outperforms it by achieving up to $55.10\%$ and $77.32\%$ accuracy on NQ Open and TriviaQA respectively.

## 1 Introduction

Recently developed retriever-reader systems (Chen et al. 2017; Karpukhin et al. 2020; Khattab and Zaharia 2020; Izacard and Grave 2021) have achieved impressive performance on the open-domain question answering task. In this pipeline, the retriever finds the top-$N$ relevant passages and the reader model leverages them to predict the answer. Prior work has shown that the performance of the reader tends to improve (up to a certain extent) with the increase in the value of $N$. Thus, state-of-the-art models use a large number of passages (e.g. 100). While this strategy results in a high prediction performance, it makes the inference of the reader computationally very expensive. For instance, Fusion-in-Decoder reader model (FiD) (Izacard and Grave 2021) requires approximately $70 \times 10^{11}$ floating-point operations (FLOPs) for an inference with $100$ passages. This high inference cost limits the widespread adoption of such systems in real-world applications that prefer efficient systems to be able to achieve low response times.

Improving the efficiency of systems has been an important research topic in NLP. For the open-domain QA (ODQA) task, efficiency from the perspectives of retrieval (Zhao, Lu, and Lee 2021) and on-disk memory (Min et al. 2021; Izacard et al. 2020; Yamada, Asai, and Hajishirzi 2021) has been studied. However, the aspect of efficiently leveraging external knowledge to improve the computation performance of the reader model has remained underexplored.

Motivated by the efficient approach that humans typically use for answering questions, we argue that the reader model does not always require all the top-$N$ passages to answer a question correctly. Some questions are trivial and can be answered with a few passages or even without using any external knowledge at all (by just relying on the knowledge already stored in the model parameters). Consider the case of FiD model, it achieves $54.43\%$ and $50.61\%$ exact match accuracies when used with $100$ and $10$ passages respectively. From this, it is clear that the performance of the system utilizing $100$ passages can be matched by inferring a large number of instances with just $10$ passages and only a few instances with $100$ passages. Furthermore, the closed-book model (Roberts, Raffel, and Shazeer 2020) that does not use any external knowledge and relies only on the knowledge in its parameters (acquired during pre-training/fine-tuning) requires considerably lesser FLOPs and achieves lower yet non-trivial accuracy of $29.83\%$. **Thus, by carefully deciding when external knowledge is required and whether the current amount of external knowledge is sufficient to answer a question correctly, the computational efficiency of the reader system can be considerably improved while**


Accepted to Workshop on Knowledge Augmented Methods for Natural Language Processing, in conjunction with AAAI 2023.

**maintaining the high prediction accuracy.** Moreover, this can also help the system mitigate the distraction that may result from using too many passages for inference and thus can even improve its prediction accuracy.

Following the above intuition, we explore an approach that utilizes both 'closed-book' and 'open-book' inferences and dynamically uses external knowledge in multiple *knowledge iterations*. Specifically, given a question, we first answer it using the low-cost closed-book model that relies on the knowledge already stored in its parameters. If its prediction confidence is sufficiently high then the prediction is outputted otherwise we use the open-book model that leverages external knowledge. Unlike the standard open-book models that always 'read' a fixed number of passages, in our method, the knowledge provided is iteratively increased until the model's prediction becomes sufficiently confident. We study four ways to measure the confidence of prediction (Section 3) of the models and demonstrate that the confidence shows a positive correlation with the predictive correctness. Thus, instances that are predicted with high confidence using low-cost inference get answered at early stages as their predictions are likely to be correct, and the remaining instances get answered with dynamically used external knowledge. Hence, by avoiding expensive inference primarily for easy instances and dynamically using external knowledge, our approach makes the reader system computationally efficient while maintaining high prediction accuracy.

Through comprehensive experiments on NQ and TriviaQA datasets, we first show that our approach considerably improves the computational efficiency of inference of the reader model. Comparing with the FiD reader, we show that our approach matches its accuracy by utilizing just $18.32\%$ of its reader inference computation cost (in FLOPs). Then, we show that our approach also leads to a consistent improvement in prediction accuracy. Specifically, it outperforms FiD by achieving up to $55.10\%$ accuracy on NQ Open and $72.33\%$ on TriviaQA. This improvement is an outcome of mitigating distraction that results from using too many passages for inference. Finally, we note that our approach is intuitive, easy to implement, and also has practical values.

## 2 Related Work

In recent times, a considerable research effort has been invested in improving the efficiency of NLP systems. It has lead to development of several techniques, such as *network pruning* (Wang, Wohlwend, and Lei 2020; Guo, Rush, and Kim 2021), *quantization* (Shen et al. 2020; Zhang et al. 2020; Tao et al. 2022), *knowledge distillation* (Clark et al. 2019; Jiao et al. 2020; Li et al. 2022; Mirzadeh et al. 2020), dynamic inference (Xin et al. 2020), adaptive model size (Goyal et al. 2020; Kim and Cho 2021; Hou et al. 2020), cascading (Xin et al. 2021; Varshney and Baral 2022), and *input reduction* (Modarressi, Mohebbi, and Pilehvar 2022).

In open-domain QA, especially for the retriever-reader systems, efficiency from the perspectives of retrieval (Zhao, Lu, and Lee 2021; Luo et al. 2022b) and on-disk memory (Min et al. 2021; Izacard et al. 2020) has been explored. However, efficiently leveraging external knowledge to improve the computation efficiency of reader inference is also important and has remained underexplored.

Our work differs from existing work in the following aspects: (1) Firstly, the inference efficiency aspect has mostly been studied for classification tasks using encoder-based models. However, we focus on a more challenging task of open-domain QA with generative models. (2) Existing efficiency improvement methods typically require architectural changes, network manipulation, saliency quantification, knowledge distillation, or even complex training procedures. In contrast, our method is easy to implement, does not require such modifications, and could generalize easily to a variety of applications. Moreover, it can even complement these existing methods. (3) The computation efficiency in existing methods often comes with a compromise on accuracy. In contrast, we show that our method consistently achieves superior accuracy. (4) Finally, existing methods usually do not allow custom computation costs and require training a separate model for each computation budget. In contrast, our system can be adjusted to meet any given computational requirements.

## 3 Approach

In open-domain QA, the cost of reader inference depends on the number of passages used as additional context for the inference. Prior work has shown that the performance of the reader tends to improve (up to a certain extent) with the increase in the number of these passages. Thus, state-of-the-art models use a large number of passages (e.g. 100). While this strategy results in a high prediction performance, it makes the inference of the reader computationally very expensive. Motivated by the strategy that humans typically use for answering questions, we explore an approach that efficiently leverages the external knowledge by dynamically using it in multiple *knowledge iterations*.

Firstly, we note that even a **closed-book** reader (CB) that does not use any external knowledge achieves a non-trivial accuracy by just relying on the knowledge already stored in its network parameters (acquired during pre-training/fine-tuning). This is a low-cost inference as the input contains just the question (without any additional context). So, in our approach, we first infer the given question using the closed-book reader and output the prediction if it is already sufficiently confident. If it is not confident then we leverage the external knowledge with the **open-book** reader (OB). Unlike the standard open-book readers that use all the top retrieved passages for inference, we iteratively increase the number of passages until the reader predicts with sufficient confidence; we refer to these iterations as '**knowledge iterations**'.

This conditional multi-stage inference process achieves computation efficiency benefits for two reasons: first, if CB reader is already sufficiently confident in its prediction then the expensive open-book inference is not used at all (this corresponds to the case of the least inference cost) and second, when OB reader is indeed used for inference, it leverages the external knowledge efficiently by reading just enough passages required to predict confidently instead of reading a static large number of passages. This approach can also

help the system mitigate the distraction that may result from using too many passages for inference.

Hence, by avoiding expensive inference and dynamically using the external knowledge, our approach makes the reader inference computationally efficient while maintaining high prediction accuracy. We note that this **doesn't impact the retriever as the retrieval is done only once** irrespective of the number of knowledge iterations and the number of passages used in each individual iteration. We further note that **in this work, our focus is only on improving the cost of reader inference**. We detail our approach and provide its mathematical formulation in the next subsection.

## Mathematical Formulation

Let $q$ be the given question, $K$ be the number of knowledge iterations, and $M^q_{OB_k}$ be the value indicating whether $k^{th}$ iteration ($k \leq K$) with the OB reader is used for inference.

$$M^q_{OB_k} = \begin{cases} 1, & \text{if } k^{th} \text{ knowledge iteration with} \\ & \text{OB reader is used for question } q \\ 0, & \text{otherwise} \end{cases}$$

**Sample Scenario:** Consider a scenario in which top-100 relevant passages are available and in our approach we are using two knowledge iterations ($K = 2$) using 20 and 100 passages in the two iterations respectively i.e. a question will be first inferred using the CB reader, if it is not sufficiently confident then top-20 (referred as $S_1$) passages will be used with the OB model (referred as $OB_1$) and if that prediction is not confident then top-100 ($S_2$) passages will be used with the OB model ($OB_2$). In the computationally most efficient case, inference would be made only using the CB reader and in the most expensive case, inference would be made sequentially using CB, then $OB_1$ (OB using 20 passages), and then $OB_2$ (OB using 100 passages). In this work, we comprehensively study extensive combinations of CB, OB readers and values of $K$ and $S_k$.

**Cost of Reader Inference:** In our general formulation, the cost of reader inference for instance $q$ is calculated as:

$$Cost^q = C_{CB} + \sum_{k=1}^{K} (M^q_{OB_k} \times S_k \times C_{OB})$$

where $C_{CB}$ and $C_{OB}$ are the inference costs of CB and OB models respectively, and $S_k$ corresponds to the number of passages used by the OB model in $k^{th}$ knowledge iteration. This is because the low-cost CB reader is first used for all the instances and then OB model is conditionally used in different knowledge iterations. In the sample scenario, the minimum cost would be equal to $C_{CB}$ and the maximum cost would be $C_{CB} + 20 \times C_{OB} + 100 \times C_{OB}$. Next, we discuss a few important characteristics of this formulation.

**Inference Cost of OB Reader:** Fusion-in-Decoder model (Izacard and Grave 2021) is one of the top performing open-book models. It uses an encoder-decoder architecture i.e. it first computes the representation of question + passage for each passage independently using the encoder (with fixed number of input tokens) and then concatenates these representations and passes it to the decoder for making the answer prediction. Thus, to compute the OB model's cost of inference in the $k^{th}$ iteration, we multiply $C_{OB}$ with $S_k$ where $C_{OB}$ is the cost of single inference with that fixed number of tokens and it is done $S_k$ number of times.

However, we note that this is the upper bound of the inference cost because the encoder representation of the passages of the previous iterations can be reused i.e. the representations of passages of $(k-1)^{th}$ iteration can be reused in the $k^{th}$ iteration. However, this would require auxiliary space for storage and would involve a trade-off between computation cost and storage space. We leave the investigation of this trade-off for future work but **note that the inference cost of our method would be even lower in practice**. Moreover, as we demonstrate through extensive experiments that even with this cost upper bound, the proposed method achieves very high improvements in computation efficiency.

**Same Model for CB and OB:** We note that the same model can also be used to act as CB when external knowledge is not available and as OB when it is available. However, their cost of inference will still be different as it depends on the number of input tokens used for inference. The CB reader uses only the question as input while the OB reader also uses the external knowledge (thus more number of input tokens). Therefore, to keep our formulation general, we keep two different variables for their respective costs.

**Total Cost of Reader Inference:** We calculate the average cost of reader inference for the evaluation dataset $D$ as:

$$Cost_D = \frac{\sum_{q_j \in D} Cost^{q_j}}{|D|}$$

## Deciding When to Use More Knowledge

The leading models in ODQA and perhaps in many NLU tasks are nowadays mostly seq2seq generative models; for e.g. both the closed-book and FiD models are based on T5 (Raffel et al. 2020). Therefore, the main design decision in our method corresponds to computing the confidence of prediction for these models. These models make their predictions token by token and output a probability distribution over the entire vocabulary for each token. We explore a number of methods to compute the confidence scores.

Let the maximum softmax probabilities for the prediction having $n$ tokens at each token position be $(p_1, p_2, p_3, ..., p_n)$. We explore the following ways of computing the model's prediction confidence using these $p$ values:

**Product of probabilities of all tokens ($P_{PA}$):** In this technique, we take the product of probabilities of all tokens of the prediction i.e. $PROD(p_1, p_2, ..., p_n)$ as the model's prediction confidence. This is the standard technique used in various tasks such as perplexity computation. We also experiment with several other confidence techniques.

**Probability of the first token ($P_F$):** In this technique, we simply use the probability of the first token of the prediction i.e. $p_1$ as the model's prediction confidence.

**Average probability of first and last token ($P_{FL}$):** Here, we use the average probability of the first and the last token i.e. $AVG(p_1, p_n)$ as the model's prediction confidence.

**Average probability across all tokens ($P_A$):** In this technique, we utilize probabilities of all the tokens of the prediction and take the average probability across all tokens i.e. $AVG(p_1, p_2, ..., p_n)$ as model's prediction confidence.

### Baseline Approaches

For a fair comparison of our approach (and various confidence computation methods), we also compare its performance with several other simple baselines:

**Random:** In this method, instead of using a metric based on probabilities to decide which instances to pass to the OB model/next knowledge iteration, we do this instance selection process at random.

**Heuristic:** Here, we use a heuristic derived from the input question to decide which instances to pass to the OB model/next knowledge iteration. Specifically, we use length of the input question as the heuristic.

### Performance Comparison Metric

We demonstrate the efficacy of our method by showing the **computation efficiency** and **accuracy** improvements. For demonstrating computation efficiency, we use FLOPs as the metric. An alternative metric could be measuring the **computation time** of inference; however, it is a machine-dependent metric. FLOPs on the other hand, is machine-independent and hence a reliable metric for comparison.

To further compare various confidence computation methods and baselines, we use another performance metric. For our approach, the reader inference cost (in FLOPs) and the accuracy vary with the prediction confidences thresholds. Therefore, to compare various confidence methods and comprehensively study their efficacy, we compute accuracies for a range of costs and plot an accuracy-cost curve (as shown in Figure 1). We plot a curve for each method and calculate the **area under the curve (AUC)** to quantify the overall performance of each method. The larger the AUC value, the better the method is as it implies higher accuracy on average across all computation costs.

## 4 Experiments and Results

**Experimental Details:** We conduct experiments with NQ (Kwiatkowski et al. 2019) Open and TriviaQA (Joshi et al. 2017) datasets. We use closed-book models from Roberts, Raffel, and Shazeer (2020) and FiD open-book models (and retrieved passages) from Izacard and Grave (2021). The closed-book models take just the question as input. The closed-book inference cost is $0.0046 \times 10^{11}$ FLOPs for T5-small and $0.0615 \times 10^{11}$ FLOPs for T5-large for the average input size (12 tokens) of NQ questions. On the other hand, the open-book readers also take the retrieved passages as input (truncated to 250 tokens each) and thus have a higher inference cost. For example, the cost of inference for the open-book FiD reader with 1 passage is $0.202 \times 10^{11}$ FLOPs for T5-base and $0.707 \times 10^{11}$ FLOPs for T5-large. We compute these and other FLOP values using 'thop' python library.

**Accuracy-Cost Curves:** As motivated previously, we plot accuracy-cost curves to study our method. We conduct experiments in multiple settings that differ in the (CB, OB) model combination, number of knowledge iterations with the OB model ($K$), and the $S_k$ values. Figure 1, 2, and 3 show these curves for $K$ = 1, 2, and 3 settings respectively. Each curve includes the following:

1. **Accuracy-cost points of individual systems**: The costs-accuracy point of the CB and $OB_k$ models are represented by the red scatter points. These individual points correspond to the case in which all the instances are answered using the corresponding reader model.
2. **Accuracy-cost curve of the proposed method**: We explore multiple ways of computing prediction confidence of the generative models. We compare the AUC (of their corresponding accuracy-cost curves) achieved by each method in Table 1. We present an exhaustive comparison of these methods in Tables 5 and 6 in Appendix. Since $\boldsymbol{P_{PA}}$ **yields the best performance**, we present accuracy-cost curves of the $P_{PA}$ method in the main results, and the other methods in the Appendix. We note that the other confidence methods also outperform the baselines.
3. **Costs at Equal Accuracies**: To measure the improvement in efficiency, we highlight (with $\times$) the costs at which the accuracy of the proposed system matches the accuracy of the $OB_k$ models. For instance, in K=1 setting, we highlight the point at which the proposed system achieves the same accuracy as the $OB_1$ model (that use a fixed $S_1$ number of passages for all questions).
4. **Accuracy-cost curve of the baseline method**: To demonstrate the efficacy of $P_{PA}$ method in comparison to the baseline method, we represent the accuracy-cost curve of the baseline with black dashed line.

In the next three subsections, we show the results for configurations with different number of knowledge iterations.

### One Knowledge Iteration (K = 1)

In this setting, we first use the CB model and if it is not sufficiently confident then we use the OB model with $S_1$ knowledge statements. Figure 1 shows the accuracy-cost curves for different configurations of CB, OB, and $S_1$ values on NQ.

**Improvement in Efficiency:** The accuracy-cost curves show that the proposed system matches the accuracy of the OB model at a considerably lesser inference cost. This cost value corresponds to the point of intersection ($\times$) on the curve with a straight horizontal line drawn from $OB_1$. For example, in the case of (CB= T5-large, OB=T5-base, $S_1 = 5$), $OB_1$ achieves $43.30\%$ accuracy at the computation cost of $1.01 \times 10^{11}$ FLOPs while the proposed system achieves the same accuracy at just $0.54 \times 10^{11}$ FLOPs. Such improvements are observed for all the cases. We show these curves for an exhaustive set of configurations in Appendix. In this setting, our approach achieves cost improvements of up to $67.77\%$. This efficiency benefit comes from using the low-cost closed-book model for some instances where it is likely to be correct and additionally using the more expensive open-book model with $S_1$ passages only for the remain-

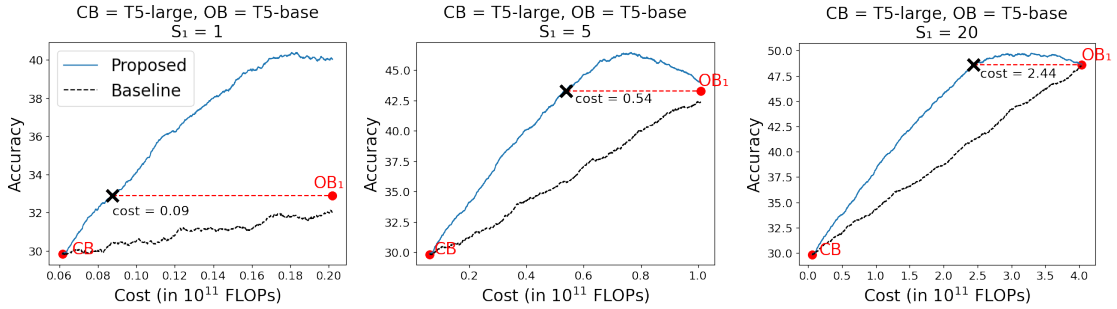


Figure 1: Accuracy-cost curves of the proposed system (in blue) and baseline (in black) for K=1 setting on NQ. Red points correspond to the accuracy and cost values of the individual models $CB$ and $OB_1$ (using $S_1$ knowledge statements). Point of intersection ($\times$) of red dashed line drawn from $OB_1$ on the blue curve corresponds to cost at which the proposed system achieves the same accuracy as $OB_1$. **Our method achieves this accuracy at considerably lower computation cost**.

| $S_1 \rightarrow$ | 1 | 2 | 5 | 10 | 20 | 50 |
|---|---|---|---|---|---|---|
| **Random** | 30.96 | 33.28 | 36.05 | 38.12 | 39.27 | 39.96 |
| **Heuristic** | 31.90 | 33.97 | 36.90 | 38.77 | 39.69 | 40.22 |
| $P_{PA}$ | **36.61** | **38.53** | **41.14** | **42.82** | **43.69** | **44.43** |
| $P_F$ | 36.15 | 38.09 | 40.77 | 42.47 | 43.34 | 44.13 |
| $P_{FL}$ | 36.13 | 38.08 | 40.76 | 42.46 | 43.35 | 44.16 |
| $P_A$ | 36.39 | 38.35 | 41.00 | 42.74 | 43.56 | 44.30 |

Table 1: Comparing AUCs of accuracy-cost curves of different confidence computation techniques for CB=T5-large, OB=T5-base configuration in K=1 setting on NQ.

ing instances. In the later results sections, we further discuss the overall cost improvement in reader inference (Table 4).

**Improvement in Accuracy:** From the accuracy-cost curves, it is clear that the accuracy achieved by the proposed system surpasses the accuracy of the $OB_1$ model beyond the cost shown with $\times$. For example, in case of (CB= T5-large, OB=T5-base, $S_1 = 1$, the top-left figure), our system achieves the accuracy of up to $40.17\%$ as compared to $32.88\%$ accuracy of $OB_1$. Same as the efficiency improvement, such accuracy improvements are also observed across all configurations (shown in Appendix). We attribute this improvement in accuracy to our approach's efficient use of external knowledge i.e. relying on the closed-book model when it is likely to be correct thus avoiding distraction with the external knowledge and using it only when it is required.

**Distraction At Inference:** The fact that our approach (that outputs its predictions using CB for some instances and OB for the others) outperforms the OB reader highlights that excessive external knowledge distracts the reader into giving incorrect answers while the CB reader answers it correctly.

We note that near the $OB_1$ computation cost, the accuracy of our system begins to come closer to the $OB_1$'s accuracy as more and more instances get answered by the $OB_1$ model.

**Comparison with Baselines:** From the accuracy-cost curves, it is clear that our proposed method that uses $P_{PA}$ as the prediction confidence (blue curve) outperforms the baseline (black curve) as the blue curve is consistently above the black curve and thus has higher AUC value. In Table 1, we compare AUC values achieved by all the confidence measures: $P_F$, $P_{FL}$, $P_A$, and $P_{PA}$. All the approaches achieve considerably higher AUCs than the baselines while $P_{PA}$ achieves the highest. This highlights the effectiveness of our confidence computation methods. We show this comparison for other configurations in Appendix. As $P_{PA}$ achieves the highest AUC, we use it for our subsequent experiments.

| Method | NQ | TQA |
|---|---|---|
| Hard EM (Min et al. 2019) | 28.8 | 50.9 |
| ORQA (Lee et al. 2018) | 31.3 | 45.1 |
| REALM (Guu et al. 2020) | 40.4 | - |
| DPR (Karpukhin et al. 2020) | 41.5 | 57.9 |
| RAG (Lewis et al. 2020) | 44.5 | 56.1 |
| DensePhrases (Lee et al. 2021) | 41.5 | 56.8 |
| PAQ (Lewis et al. 2021) | 52.3 | - |
| KG-FiD (Yu et al. 2022) | 53.4 | 69.8 |
| FiD* (Izacard and Grave 2021) (base) | 50.03 | 68.01 |
| **Ours K= 1 (with FiD base)** | **50.83** | **68.52** |
| **Ours K= 2 (with FiD base)** | **50.94** | **68.69** |
| **Ours K= 3 (with FiD base)** | **50.97** | **68.80** |
| FiD* (Izacard and Grave 2021) (large) | 54.43 | 72.07 |
| **Ours K= 1 (with FiD large)** | **54.90** | **72.16** |
| **Ours K= 2 (with FiD large)** | **54.99** | **72.29** |
| **Ours K= 3 (with FiD large)** | **55.10** | **72.33** |

Table 2: Comparing EM performance of ODQA methods. * indicates the highest performance of the latest model.

**Comparing Overall Performance:** We show the exact match accuracies achieved by various ODQA methods in Table 2. Our method achieves slightly higher performance that both FiD base and large models. We note that this is an additional benefit of our approach, though, the primary benefit remains to be the improvement in efficiency. The cost of FiD base model is $20.19 \times 10^{11}$ FLOPs while our system matches its accuracy at just $13.10 \times 10^{11}$ FLOPs. We show that the performance improves further on increasing the number of knowledge iterations in our method.

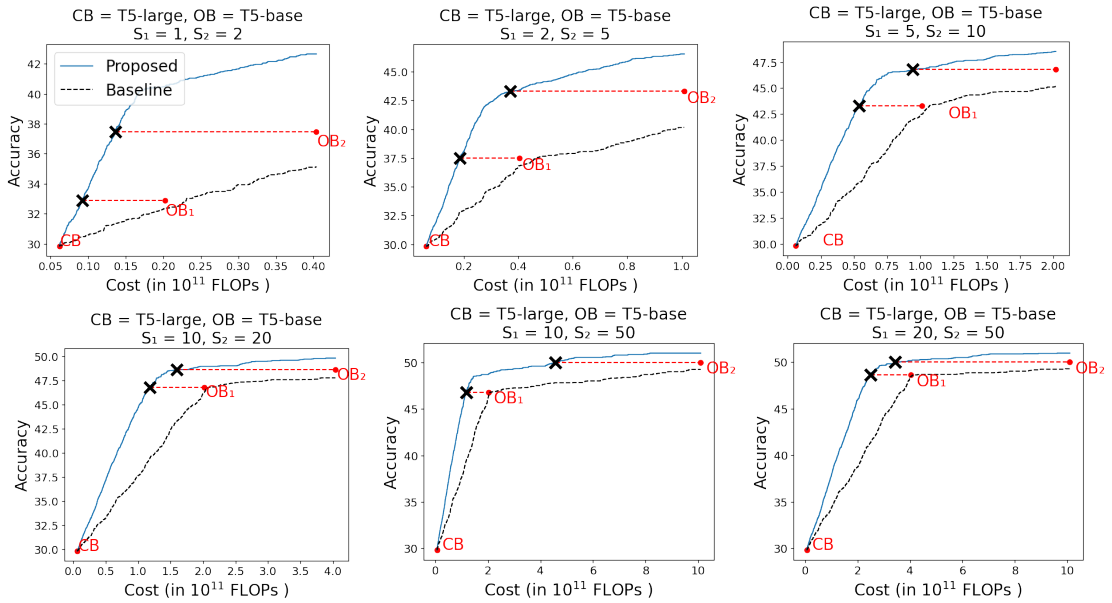


Figure 2: Accuracy-cost curves of the proposed method (in blue) and baseline (in black) for K=2 setting on NQ. Red points correspond to the accuracy and cost values of the individual models $CB$, $OB_1$ (using $S_1$ knowledge), and $OB_2$ (using $S_2$ knowledge). Points of intersection of red dashed lines drawn from $OB_1$ and $OB_2$ on the blue curve correspond to costs at which our system achieves the same accuracy as $OB_1$ and $OB_2$ respectively.

| CB | OB | $S_1$ | $S_2$ | $P_A$ | $P_{PA}$ | Baseline |
|---|---|---|---|---|---|---|
| T5-Large | T5-Base | 1 | 2 | 39.56 | **39.79** | 32.75 |
| T5-Large | T5-Base | 2 | 5 | 42.90 | **43.14** | 36.62 |
| T5-Large | T5-Base | 10 | 20 | 46.23 | **46.30** | 42.83 |
| T5-Large | T5-Base | 20 | 100 | 49.04 | **49.09** | 47.07 |
| T5-Large | T5-Large | 1 | 2 | 42.31 | **42.48** | 35.76 |
| T5-Large | T5-Large | 2 | 5 | 46.24 | **46.35** | 41.08 |
| T5-Large | T5-Large | 10 | 20 | 49.55 | **49.71** | 46.51 |
| T5-Large | T5-Large | 20 | 100 | 52.87 | **52.97** | 51.44 |

Table 3: Comparing AUCs of accuracy-cost curves of the proposed and the baseline methods in K=2 setting.

## Two Knowledge Iterations (K = 2)

In this setting, we use CB model and then conditionally use two knowledge iterations with the OB model. Specifically, if CB is not sufficiently confident then we use the OB model with $S_1$ passages ($OB_1$), and if $OB_1$ is not sufficiently confident then we use the OB model with $S_2$ passages ($OB_2$). Figure 2 shows accuracy-cost curves for this setting.

**Improvement in Efficiency:** In this setting, our method achieves larger efficiency improvements than the K=1 setting. For example, in case of (CB=T5-large, OB=T5-base, $S_1$=1, $S_2$=2), $OB_2$ achieves $37.48\%$ accuracy at the cost of $0.4 \times 10^{11}$ FLOPs and cascading system achieves the same accuracy at just $0.13 \times 10^{11}$ FLOPs. We achieve similar efficiency improvements over $OB_1$ model also; in the same case, $OB_1$ achieves $32.88\%$ accuracy at $0.2 \times 10^{11}$ FLOPs and our system achieves the same accuracy at just $0.09 \times 10^{11}$ FLOPs. Similar improvements are observed for all the configurations as we show in Appendix.

**Improvement in Accuracy:** As can be observed from the accuracy-cost curves, our system achieves a higher accuracy than even the $OB_2$ model. For example, in case of (CB=T5-large, OB=T5-base, $S_1$=10, $S_2$=20), our system achieves accuracy of $49.83\%$ that is considerably higher than the $48.61\%$ accuracy of $OB_2$ and $46.78\%$ accuracy of $OB_1$. Furthermore, at the same cost as $OB_1$, cascading system achieves $48.98\%$ accuracy, $2.2\%$ higher than that of $OB_1$ system ($46.78\%$). Thus, our method improves both the reader computation efficiency and the prediction accuracy.

**Comparison with Baseline:** In Table 3, we show AUCs for different configurations. Same as K=1 setting, the proposed system clearly achieves higher AUC than the baseline.

## Three Knowledge Iterations (K = 3)

In this setting, we first use the CB model and then conditionally use three knowledge iterations with the OB model with $S_1$, $S_2$, and $S_3$ passages respectively. Figure 3 shows the accuracy-cost curves for different configurations.

**Improvement in Efficiency and Accuracy:** We observe both efficiency and accuracy improvements in this setting also. For example, in (CB=T5-large, OB=T5-base, $S_1$=10, $S_2$=20, $S_3$=100) configuration, $OB_3$ achieves $50.03\%$ accuracy at cost of $20.19 \times 10^{11}$ FLOPs and our system achieves the same accuracy at just $5.03 \times 10^{11}$ FLOPs. Furthermore, our system even achieves higher accuracy than even $OB_3$. Finally, the proposed system achieves AUC of $49.62$ which is considerably higher than that of the baseline ($47.7$).

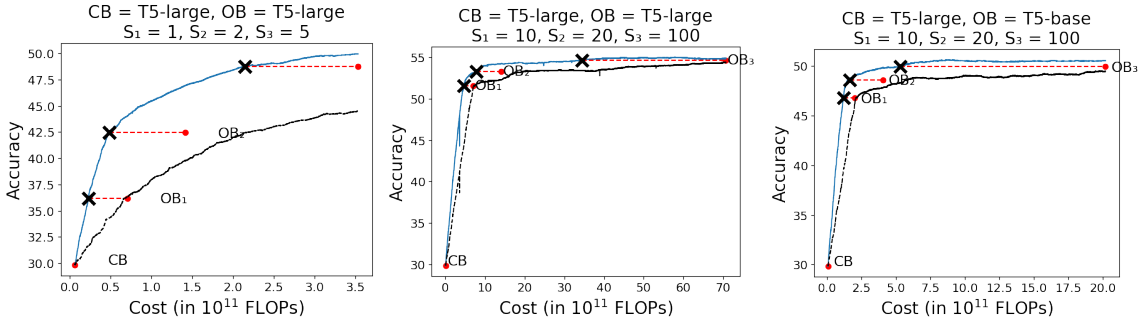


Figure 3: Accuracy-cost curves of the proposed method (in blue) and baseline (in black) for K=3 setting on NQ. Red points correspond to the accuracy and cost values of the individual $CB$, $OB_1$, $OB_2$, and $OB_3$ models.

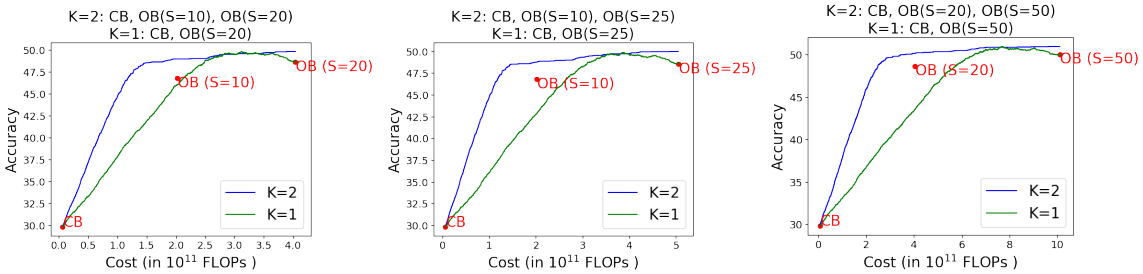


Figure 4: [Best viewed in color] Illustrating the **impact of multiple knowledge iterations** by plotting the accuracy-cost curves for K=1 and K=2 settings together. The system using two iterations (K=2) achieves higher AUC than its counterpart using the same amount of total knowledge (20, 25, and 50 in the three cases respectively) but with just one iteration.

| **Method** | **Cost (in $10^{11}$ FLOPs)** |
|---|---|
| FiD (base) | 20.19 |
| Ours (at same EM as FiD base) | 3.69 |
| FiD (large) | 70.69 |
| Ours (at same EM as FiD large) | 20.59 |

Table 4: Comparing reader inference cost of FiD and our system at equivalent exact match accuracies on NQ.

## 5 Impact of Knowledge Iterations

We demonstrate the impact of multiple knowledge iterations by plotting the accuracy-cost curves for K=1 and K=2 settings together in Figure 4. The system using two iterations (K=2) achieves higher AUC than its counterpart using the same amount of knowledge but with just one iteration. For the first case, we use CB and OB model (with S=20) in K=1 setting, and in the K=2 setting, we introduce an intermediate step that uses 10 passages i.e. we use CB, $OB_1$ (with $S_1$=10), and $OB_2$ (with $S_2$=20). The total amount of knowledge is same in both the scenarios (20 contexts) but K=2 system tries to first answer the question using just 10 passages while the K=1 system directly uses 20 passages. The K=2 system achieves higher AUC than K=1 (46.25 vs 43.56). This pattern is seen in all the cases thus highlighting the positive impact of using knowledge iterations with OB model.

## 6 Comparing Overall Performance

In Table 2, we compare the performance achieved by our system in different K settings. With the increase in the value of K, the improvement in performance also increases. On NQ Open, our system achieves accuracy of up to $55.10\%$ with large model and up to $50.97\%$ with base model outperforming all other reader methods such as Hard EM, ORQA, REALM, DPR, RAG, DensePhrases, PAQ, FiD, and KG-FiD. Similar improvements are also observed on the TriviaQA. Finally, in Table 4, we compare the cost of reader inference at equal EMs and show that our method achieves considerable efficiency improvements over the FiD reader.

## 7 Conclusion

Addressing the problem of high-inference cost of reader models in open-domain QA, we explored an approach that utilizes both the 'closed-book' and the 'open-book' inference and dynamically reads the external knowledge in multiple 'knowledge iterations'. Through comprehensive experiments, we demonstrated that this dynamic reading approach improves both the **inference efficiency** and the **prediction accuracy** of the reader. Compared with the top-performing Fusion-in-Decoder reader, this approach matches its accuracy by utilizing just $18.32\%$ of its reader inference cost (FLOPs) and also outperforms it by achieving up to $55.10\%$ accuracy on NQ Open and $72.33\%$ on TriviaQA. Finally, we hope that our work will encourage further research and facilitate development of efficient QA reader systems.

## References


Ben Zaken, E.; Goldberg, Y.; and Ravfogel, S. 2022. BitFit: Simple Parameter-efficient Fine-tuning for Transformer-based Masked Language-models. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 1–9. Dublin, Ireland: Association for Computational Linguistics.

Chen, D.; Fisch, A.; Weston, J.; and Bordes, A. 2017. Reading Wikipedia to Answer Open-Domain Questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1870–1879. Vancouver, Canada: Association for Computational Linguistics.

Clark, K.; Luong, M.-T.; Khandelwal, U.; Manning, C. D.; and Le, Q. V. 2019. BAM! Born-Again Multi-Task Networks for Natural Language Understanding. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 5931–5937. Florence, Italy: Association for Computational Linguistics.

Garg, S.; and Moschitti, A. 2021. Will this Question be Answered? Question Filtering via Answer Model Distillation for Efficient Question Answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, 7329–7346. Online and Punta Cana, Dominican Republic: Association for Computational Linguistics.

Goyal, S.; Choudhury, A. R.; Raje, S.; Chakaravarthy, V.; Sabharwal, Y.; and Verma, A. 2020. PoWER-BERT: Accelerating BERT inference via progressive word-vector elimination. In *International Conference on Machine Learning*, 3690–3699. PMLR.

Guo, D.; Rush, A.; and Kim, Y. 2021. Parameter-Efficient Transfer Learning with Diff Pruning. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 4884–4896. Online: Association for Computational Linguistics.

Guu, K.; Lee, K.; Tung, Z.; Pasupat, P.; and Chang, M. 2020. Retrieval Augmented Language Model Pre-Training. In III, H. D.; and Singh, A., eds., *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, 3929–3938. PMLR.

Hou, L.; Huang, Z.; Shang, L.; Jiang, X.; Chen, X.; and Liu, Q. 2020. DynaBERT: Dynamic BERT with Adaptive Width and Depth. In Larochelle, H.; Ranzato, M.; Hadsell, R.; Balcan, M. F.; and Lin, H., eds., *Advances in Neural Information Processing Systems*, volume 33, 9782–9793. Curran Associates, Inc.

Houlsby, N.; Giurgiu, A.; Jastrzebski, S.; Morrone, B.; De Laroussilhe, Q.; Gesmundo, A.; Attariyan, M.; and Gelly, S. 2019. Parameter-Efficient Transfer Learning for NLP. In Chaudhuri, K.; and Salakhutdinov, R., eds., *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, 2790–2799. PMLR.

Izacard, G.; and Grave, E. 2021. Leveraging Passage Retrieval with Generative Models for Open Domain Question Answering. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 874–880. Online: Association for Computational Linguistics.

Izacard, G.; Petroni, F.; Hosseini, L.; De Cao, N.; Riedel, S.; and Grave, E. 2020. A memory efficient baseline for open domain question answering. *arXiv preprint arXiv:2012.15156*.

Jiao, X.; Yin, Y.; Shang, L.; Jiang, X.; Chen, X.; Li, L.; Wang, F.; and Liu, Q. 2020. TinyBERT: Distilling BERT for Natural Language Understanding. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, 4163–4174. Online: Association for Computational Linguistics.

Joshi, M.; Choi, E.; Weld, D.; and Zettlemoyer, L. 2017. TriviaQA: A Large Scale Distantly Supervised Challenge Dataset for Reading Comprehension. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 1601–1611. Vancouver, Canada: Association for Computational Linguistics.

Kamath, A.; Jia, R.; and Liang, P. 2020. Selective Question Answering under Domain Shift. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 5684–5696. Online: Association for Computational Linguistics.

Karpukhin, V.; Oguz, B.; Min, S.; Lewis, P.; Wu, L.; Edunov, S.; Chen, D.; and Yih, W.-t. 2020. Dense Passage Retrieval for Open-Domain Question Answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6769–6781. Online: Association for Computational Linguistics.

Khattab, O.; and Zaharia, M. 2020. ColBERT: Efficient and Effective Passage Search via Contextualized Late Interaction over BERT. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '20, 39–48. New York, NY, USA: Association for Computing Machinery. ISBN 9781450380164.

Kim, G.; and Cho, K. 2021. Length-Adaptive Transformer: Train Once with Length Drop, Use Anytime with Search. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 6501–6511. Online: Association for Computational Linguistics.

Kwiatkowski, T.; Palomaki, J.; Redfield, O.; Collins, M.; Parikh, A.; Alberti, C.; Epstein, D.; Polosukhin, I.; Devlin, J.; Lee, K.; et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7: 453–466.

Lee, J.; Sung, M.; Kang, J.; and Chen, D. 2021. Learning Dense Representations of Phrases at Scale. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long*

*Papers)*, 6634–6647. Online: Association for Computational Linguistics.

Lee, J.; Yun, S.; Kim, H.; Ko, M.; and Kang, J. 2018. Ranking Paragraphs for Improving Answer Recall in Open-Domain Question Answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, 565–569. Brussels, Belgium: Association for Computational Linguistics.

Lewis, P.; Denoyer, L.; and Riedel, S. 2019. Unsupervised Question Answering by Cloze Translation. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, 4896–4910. Florence, Italy: Association for Computational Linguistics.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; Riedel, S.; and Kiela, D. 2020. Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20. Red Hook, NY, USA: Curran Associates Inc. ISBN 9781713829546.

Lewis, P.; Wu, Y.; Liu, L.; Minervini, P.; Küttler, H.; Piktus, A.; Stenetorp, P.; and Riedel, S. 2021. PAQ: 65 Million Probably-Asked Questions and What You Can Do With Them. *Transactions of the Association for Computational Linguistics*, 9: 1098–1115.

Li, X. L.; and Liang, P. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 4582–4597. Online: Association for Computational Linguistics.

Li, Z.; Wang, Z.; Tan, M.; Nallapati, R.; Bhatia, P.; Arnold, A.; Xiang, B.; and Roth, D. 2022. DQ-BART: Efficient Sequence-to-Sequence Model via Joint Distillation and Quantization. *arXiv preprint arXiv:2203.11239*.

Luo, M.; Hashimoto, K.; Yavuz, S.; Liu, Z.; Baral, C.; and Zhou, Y. 2022a. Choose Your QA Model Wisely: A Systematic Study of Generative and Extractive Readers for Question Answering. In *Proceedings of the 1st Workshop on Semiparametric Methods in NLP: Decoupling Logic from Knowledge*, 7–22.

Luo, M.; Jain, S.; Gupta, A.; Einolghozati, A.; Oguz, B.; Chatterjee, D.; Chen, X.; Baral, C.; and Heidari, P. 2022b. A Study on the Efficiency and Generalization of Light Hybrid Retrievers. *arXiv preprint arXiv:2210.01371*.

Min, S.; Boyd-Graber, J.; Alberti, C.; Chen, D.; Choi, E.; Collins, M.; Guu, K.; Hajishirzi, H.; Lee, K.; Palomaki, J.; Raffel, C.; Roberts, A.; Kwiatkowski, T.; Lewis, P.; Wu, Y.; Küttler, H.; Liu, L.; Minervini, P.; Stenetorp, P.; Riedel, S.; Yang, S.; Seo, M.; Izacard, G.; Petroni, F.; Hosseini, L.; Cao, N. D.; Grave, E.; Yamada, I.; Shimaoka, S.; Suzuki, M.; Miyawaki, S.; Sato, S.; Takahashi, R.; Suzuki, J.; Fajcik, M.; Docekal, M.; Ondrej, K.; Smrz, P.; Cheng, H.; Shen, Y.; Liu, X.; He, P.; Chen, W.; Gao, J.; Oguz, B.; Chen, X.; Karpukhin, V.; Peshterliev, S.; Okhonko, D.; Schlichtkrull, M.; Gupta, S.; Mehdad, Y.; and Yih, W.-t. 2021. NeurIPS 2020 EfficientQA Competition: Systems, Analyses and Lessons Learned. In Escalante, H. J.; and Hofmann, K., eds., *Proceedings of the NeurIPS 2020 Competition and Demonstration Track*, volume 133 of *Proceedings of Machine Learning Research*, 86–111. PMLR.

Min, S.; Chen, D.; Hajishirzi, H.; and Zettlemoyer, L. 2019. A Discrete Hard EM Approach for Weakly Supervised Question Answering. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, 2851–2864. Hong Kong, China: Association for Computational Linguistics.

Mirzadeh, S. I.; Farajtabar, M.; Li, A.; Levine, N.; Matsukawa, A.; and Ghasemzadeh, H. 2020. Improved knowledge distillation via teacher assistant. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 5191–5198.

Modarressi, A.; Mohebbi, H.; and Pilehvar, M. T. 2022. AdapLeR: Speeding up Inference by Adaptive Length Reduction. *arXiv preprint arXiv:2203.08991*.

Raffel, C.; Shazeer, N.; Roberts, A.; Lee, K.; Narang, S.; Matena, M.; Zhou, Y.; Li, W.; and Liu, P. J. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. *Journal of Machine Learning Research*, 21(140): 1–67.

Roberts, A.; Raffel, C.; and Shazeer, N. 2020. How Much Knowledge Can You Pack Into the Parameters of a Language Model? In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 5418–5426. Online: Association for Computational Linguistics.

Rodriguez, P.; Barrow, J.; Hoyle, A. M.; Lalor, J. P.; Jia, R.; and Boyd-Graber, J. 2021. Evaluation Examples are not Equally Informative: How should that change NLP Leaderboards? In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 4486–4503. Online: Association for Computational Linguistics.

Schick, T.; and Schütze, H. 2021. Exploiting Cloze-Questions for Few-Shot Text Classification and Natural Language Inference. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, 255–269. Online: Association for Computational Linguistics.

Shen, S.; Dong, Z.; Ye, J.; Ma, L.; Yao, Z.; Gholami, A.; Mahoney, M. W.; and Keutzer, K. 2020. Q-bert: Hessian based ultra low precision quantization of bert. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, 8815–8821.

Tao, C.; Hou, L.; Zhang, W.; Shang, L.; Jiang, X.; Liu, Q.; Luo, P.; and Wong, N. 2022. Compression of Generative Pre-trained Language Models via Quantization. *arXiv preprint arXiv:2203.10705*.

Varshney, N.; Banerjee, P.; Gokhale, T.; and Baral, C. 2022. Unsupervised Natural Language Inference Using PHL

Triplet Generation. In *Findings of the Association for Computational Linguistics: ACL 2022*, 2003–2016. Dublin, Ireland: Association for Computational Linguistics.

Varshney, N.; and Baral, C. 2022. Model Cascading: Towards Jointly Improving Efficiency and Accuracy of NLP Systems. *arXiv preprint arXiv:2210.05528*.

Varshney, N.; Mishra, S.; and Baral, C. 2022a. ILDAE: Instance-Level Difficulty Analysis of Evaluation Data. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 3412–3425. Dublin, Ireland: Association for Computational Linguistics.

Varshney, N.; Mishra, S.; and Baral, C. 2022b. Investigating Selective Prediction Approaches Across Several Tasks in IID, OOD, and Adversarial Settings. In *Findings of the Association for Computational Linguistics: ACL 2022*, 1995–2002. Dublin, Ireland: Association for Computational Linguistics.

Varshney, N.; Mishra, S.; and Baral, C. 2022c. Towards Improving Selective Prediction Ability of NLP Systems. In *Proceedings of the 7th Workshop on Representation Learning for NLP*, 221–226. Dublin, Ireland: Association for Computational Linguistics.

Wang, Z.; Wohlwend, J.; and Lei, T. 2020. Structured Pruning of Large Language Models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 6151–6162. Online: Association for Computational Linguistics.

Wang, Z.; Yu, A. W.; Firat, O.; and Cao, Y. 2021. Towards Zero-Label Language Learning. *ArXiv*, abs/2109.09193.

Xin, J.; Tang, R.; Lee, J.; Yu, Y.; and Lin, J. 2020. DeeBERT: Dynamic Early Exiting for Accelerating BERT Inference. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2246–2251. Online: Association for Computational Linguistics.

Xin, J.; Tang, R.; Yu, Y.; and Lin, J. 2021. The Art of Abstention: Selective Prediction and Error Regularization for Natural Language Processing. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, 1040–1051. Online: Association for Computational Linguistics.

Yamada, I.; Asai, A.; and Hajishirzi, H. 2021. Efficient Passage Retrieval with Hashing for Open-domain Question Answering. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*, 979–986. Online: Association for Computational Linguistics.

Yu, D.; Zhu, C.; Fang, Y.; Yu, W.; Wang, S.; Xu, Y.; Ren, X.; Yang, Y.; and Zeng, M. 2022. KG-FiD: Infusing Knowledge Graph in Fusion-in-Decoder for Open-Domain Question Answering. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, 4961–4974. Dublin, Ireland: Association for Computational Linguistics.

Zhang, W.; Hou, L.; Yin, Y.; Shang, L.; Chen, X.; Jiang, X.; and Liu, Q. 2020. TernaryBERT: Distillation-aware Ultra-low Bit BERT. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, 509–521. Online: Association for Computational Linguistics.

Zhao, T.; Lu, X.; and Lee, K. 2021. SPARTA: Efficient Open-Domain Question Answering via Sparse Transformer Matching Retrieval. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 565–575. Online: Association for Computational Linguistics.

# Appendix

## A One Knowledge Iteration (K=1)

In this setting, we first use the CB model and if it is not sufficiently confident then we use the OB model with $S_1$ knowledge statements. Figure 5 and 6 show accuracy-cost curves of different configurations for this setting.

**Improvement in Efficiency:** These curves show that the cascading system matches the accuracy of the OB model at a much lesser computation cost. This cost value corresponds to the point of intersection on the curve with a straight horizontal line drawn from $OB_1$ (red dashed line). For example, in the case of (CB= T5-large, OB=T5-base, $S_1 = 5$), $OB_1$ achieves $43.30\%$ accuracy at the computation cost of $1.01 \times 10^{11}$ FLOPs while the cascading system achieves the same accuracy at the cost of just $0.54 \times 10^{11}$ FLOPs and in the case of (CB= T5-large, OB=T5-base, $S_1 = 10$), $OB_1$ achieves $46.79\%$ accuracy at the computation cost of $2.02 \times 10^{11}$ FLOPs while the cascading system achieves the same accuracy at the cost of just $1.18 \times 10^{11}$ FLOPs. Such improvements are observed for all the cases. This efficiency benefit comes from using the low-cost closed-book model for some instances where it is likely to be correct and using the more expensive open-book model only for the other instances.

**Comparison of Methods:** From the accuracy-cost curves, it is clear that our cascading method that uses $P_A$ as the prediction confidence (blue curve) outperforms the random baseline (black curve) as it has a higher area under its curve. The demonstrates the effectiveness of the way we compute the confidence scores and utilize them to build a cascading reader. In Table 5 and 6, we compare the AUC values achieved by baseline and different cascading methods using $P_A$, $P_F$, and $P_{FL}$ as prediction confidences. All the approaches considerably higher AUCs than the baseline. $P_A$ achieves the highest AUC in all the cases.

## B Two Knowledge Iterations (K=2)

In this setting, we first infer using the CB model and then conditionally use two knowledge iterations with the open-book model. In other words, if the CB model is not sufficiently confident then we use the OB model with $S_1$ knowledge statements ($OB_1$), and if $OB_1$ is not sufficiently confident then we use the OB model with $S_2$ knowledge statements ($OB_2$). Figure 7 and Figure 8 show accuracy-cost curves of different configurations for this setting.

In this setting, our method achieves much larger efficiency improvements than the K=1 setting. For example, in case of (CB=T5-large, OB=T5-base, $S_1$=1, $S_2$=2), $OB_2$ achieves $37.48\%$ accuracy at the cost of $0.4 \times 10^{11}$ FLOPs and cascading system achieves the same accuracy at just $0.13 \times 10^{11}$ FLOPs. Moreover, we achieve efficiency improvements over $OB_1$ model also; in the same case, $OB_1$ achieves $32.88\%$ accuracy at $0.2 \times 10^{11}$ FLOPs and cascading system achieves the same accuracy at just $0.09 \times 10^{11}$ FLOPs.

In case of (CB=T5-large, OB=T5-base, $S_1$=10, $S_2$=20), cascading system achieves the same accuracy as $OB_2$ at just $39.39\%$ of $OB_2$'s computation cost. Specifically, $OB_2$ achieves $48.61\%$ accuracy at the cost of $4.04 \times 10^{11}$ FLOPs and cascading system achieves the same accuracy at just $1.59 \times 10^{11}$ FLOPs. Our method achieves efficiency improvements over $OB_1$ model also; in the above case, $OB_1$ achieves $46.78\%$ accuracy at the cost of $2.019 \times 10^{11}$ FLOPs and cascading system achieves the same accuracy at just $1.17 \times 10^{11}$ FLOPs. Similarly, in case of (CB=T5-large, OB=T5-base, $S_1$=20, $S_2$=50), cascading system achieves the same accuracy as $OB_2$ at just $33.97\%$ of $OB_2$'s cost.

## C Re-using encoded representations of previous knowledge iterations

Fusion-in-Decoder model (Izacard and Grave 2021) is one of the top performing open-book models. It first computes the representation of question + knowledge for each knowledge statement (with a fixed number of tokens) independently using the encoder and then concatenates these representations and passes it to the decoder for making the prediction. As mentioned before, the encoded representations of knowledge of $(k-1)^{th}$ iteration can be stored and re-used in $k^{th}$ knowledge iteration. This implies that only the new knowledge needs to be encoded in the next iteration. This will further improve the computational efficiency. However, it requires auxiliary space for storage. We will explore this trade-off between auxiliary space (for storing encodings of knowledge) and inference cost as a future work.

## D Related Work on Efficiency in NLP

With the introduction of large-scale pre-trained language models, the efficiency topic has attracted a lot of research attention. Efficiency is being studied from diverse lenses such as training data efficiency (Lewis, Denoyer, and Riedel 2019; Schick and Schütze 2021; Varshney et al. 2022; Wang et al. 2021; Ben Zaken, Goldberg, and Ravfogel 2022), evaluation efficiency (Rodriguez et al. 2021; Varshney, Mishra, and Baral 2022a), parameter tuning efficiency (Li and Liang 2021; Houlsby et al. 2019), retrieval efficiency (Zhao, Lu, and Lee 2021; Luo et al. 2022a,b), on-disk memory efficiency (Min et al. 2021; Izacard et al. 2020), and inference efficiency. In this work, we focus on inference efficiency of ODQA reader and propose an approach that utilizes both the 'closed-book' (relying on the knowledge already present in the model parameters) and the 'open-book' inference (leveraging external knowledge). Furthermore, instead of using a large fixed number of passages for open-book inference, we dynamically read the external knowledge in multiple 'knowledge iterations'. Our method is also related to selective prediction (Kamath, Jia, and Liang 2020; Varshney, Mishra, and Baral 2022b,c; Garg and Moschitti 2021; Xin et al. 2021) as the system first tries to answer using the low-cost inference and then selectively utilizes the high-cost inference only when it is not sufficiently confident in its prediction. Thus, by carefully deciding when external knowledge is required and whether the current amount of external knowledge is sufficient to answer a question correctly, the computational efficiency of the reader system can be considerably improved while maintaining the high prediction accuracy.

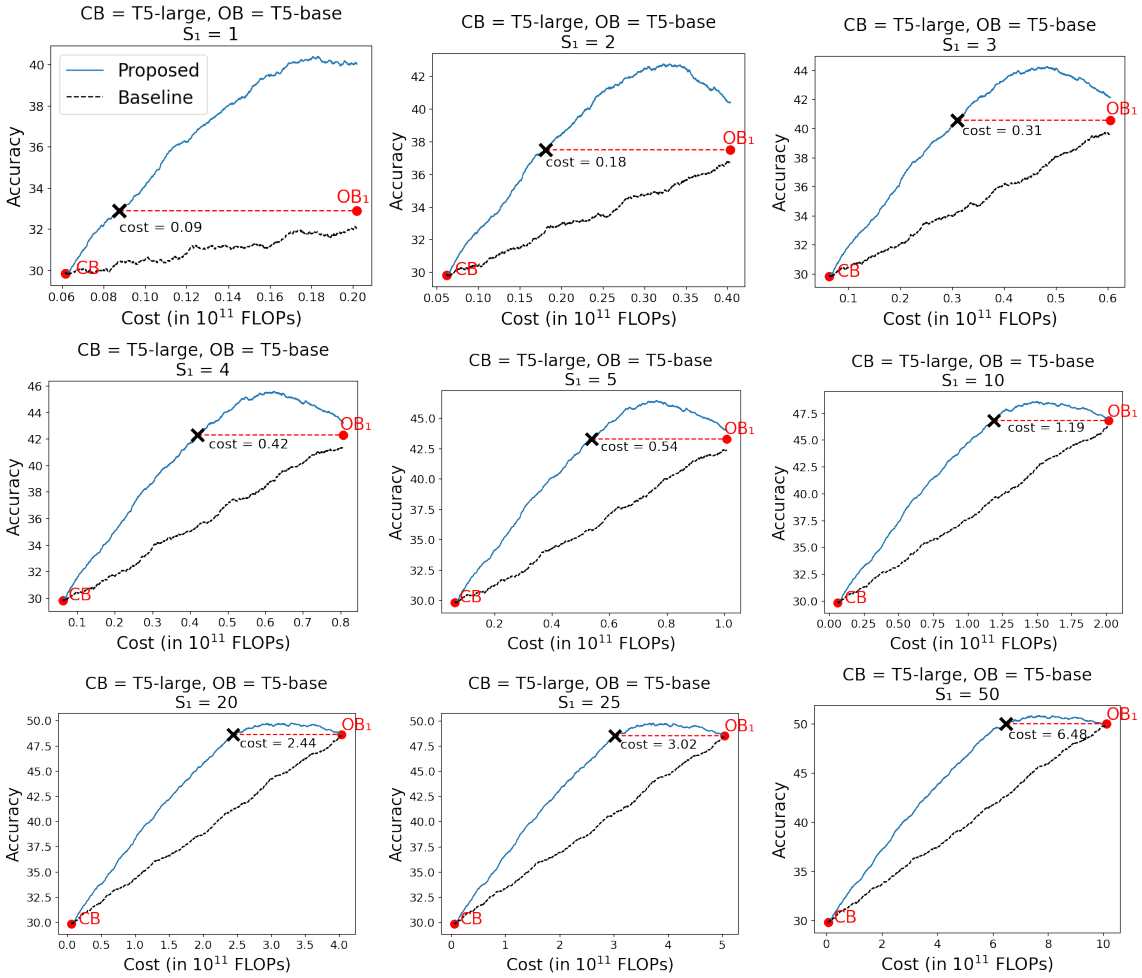


Figure 5: Accuracy-cost curves of the proposed cascading system (in blue) and baseline system (in black) for K=1 setting on NQ. Red points correspond to the accuracy and cost values of the individual models $CB$ and $OB_1$ (leveraging $S_1$ knowledge statements). Point of intersection of red dashed line drawn from $OB_1$ on the blue curve correspond to cost at which the cascading system achieves the same accuracy as $OB_1$.

| **Method \ $S_1$** | **1** | **2** | **3** | **4** | **5** | **10** | **20** | **25** | **50** | **100** |
|---|---|---|---|---|---|---|---|---|---|---|
| **Random** | 30.96 | 33.28 | 34.73 | 35.74 | 36.05 | 38.12 | 39.27 | 39.14 | 39.96 | 39.94 |
| **Heuristic** | 31.90 | 33.97 | 35.62 | 36.32 | 36.90 | 38.77 | 39.69 | 39.62 | 40.22 | 40.28 |
| $\boldsymbol{P_{PA}}$ | **36.61** | **38.53** | **39.64** | **40.59** | **41.14** | **42.82** | **43.69** | **43.69** | **44.43** | **44.29** |
| $\boldsymbol{P_F}$ | 36.15 | 38.09 | 39.16 | 40.15 | 40.77 | 42.47 | 43.34 | 43.36 | 44.13 | 43.95 |
| $\boldsymbol{P_{FL}}$ | 36.13 | 38.08 | 39.15 | 40.16 | 40.76 | 42.46 | 43.35 | 43.36 | 44.16 | 43.97 |
| $\boldsymbol{P_A}$ | 36.39 | 38.35 | 39.49 | 40.45 | 41.00 | 42.74 | 43.56 | 43.54 | 44.30 | 44.15 |

Table 5: Comparing AUCs of accuracy-cost curves of different cascading techniques for (CB=T5-large and OB=T5-base) configuration.

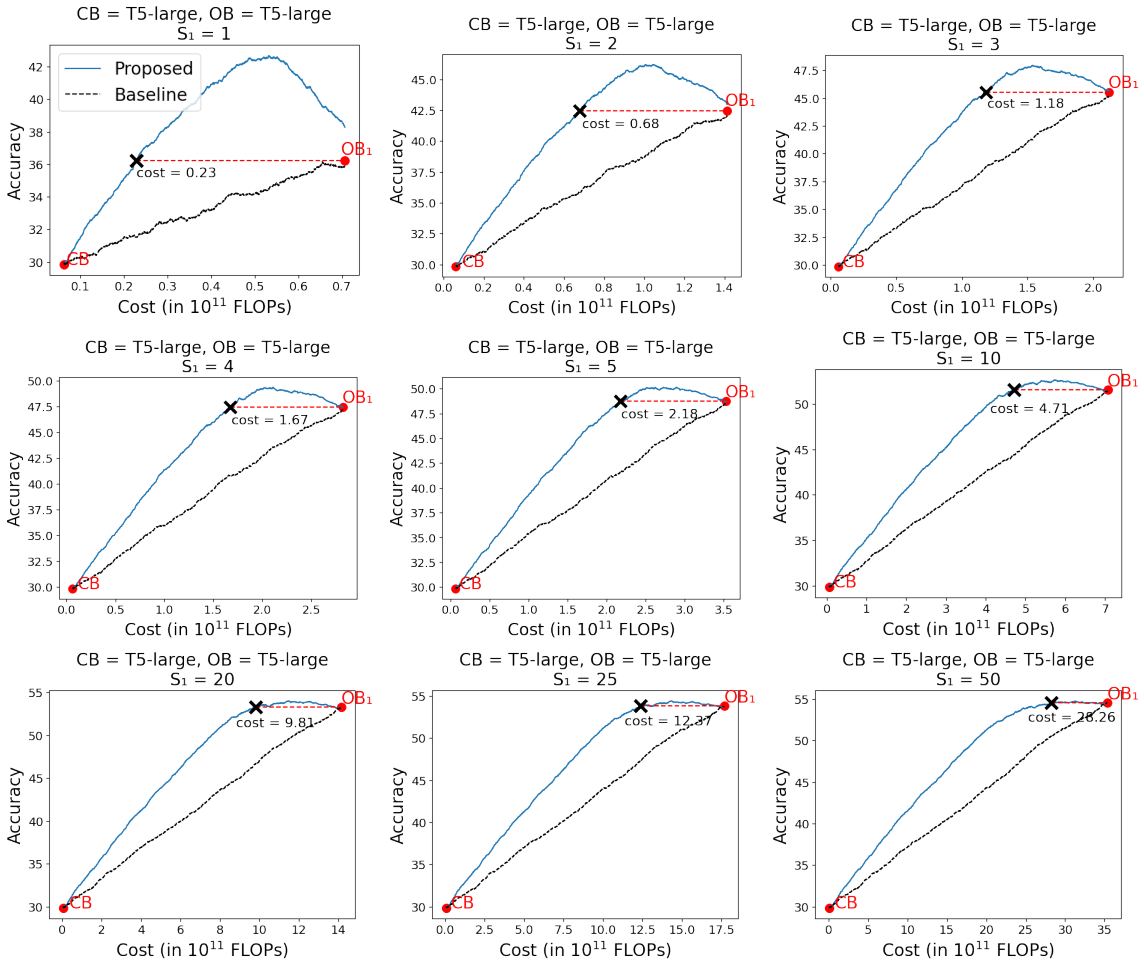


Figure 6: Accuracy-cost curves of the proposed cascading system (in blue) and baseline system (in black) for K=1 setting on NQ. Red points correspond to the accuracy and cost values of the individual models $CB$ and $OB_1$ (leveraging $S_1$ knowledge statements). Point of intersection of red dashed line drawn from $OB_1$ on the blue curve correspond to cost at which the cascading system achieves the same accuracy as $OB_1$.

| **Method \ $S_1$** | **1** | **2** | **3** | **4** | **5** | **10** | **20** | **25** | **50** | **100** |
|---|---|---|---|---|---|---|---|---|---|---|
| **Random** | 33.14 | 36.4 | 37.81 | 38.87 | 39.47 | 40.92 | 41.91 | 42.21 | 42.54 | 42.62 |
| $\boldsymbol{P_{PA}}$ | **38.45** | **41.07** | **42.29** | **43.18** | **43.76** | **45.40** | **46.34** | **46.52** | **46.71** | **46.77** |
| $\boldsymbol{P_F}$ | 38.00 | 40.68 | 41.92 | 42.87 | 43.47 | 45.11 | 46.09 | 46.31 | 46.45 | 46.46 |
| $\boldsymbol{P_{FL}}$ | 37.96 | 40.65 | 41.89 | 42.85 | 43.44 | 45.09 | 46.06 | 46.28 | 46.44 | 46.46 |
| $\boldsymbol{P_A}$ | 38.26 | 40.93 | 42.17 | 43.08 | 43.66 | 45.28 | 46.25 | 46.43 | 46.61 | 46.69 |

Table 6: Comparing AUCs of accuracy-cost curves of different cascading techniques for (CB=T5-large and OB=T5-large) configuration.

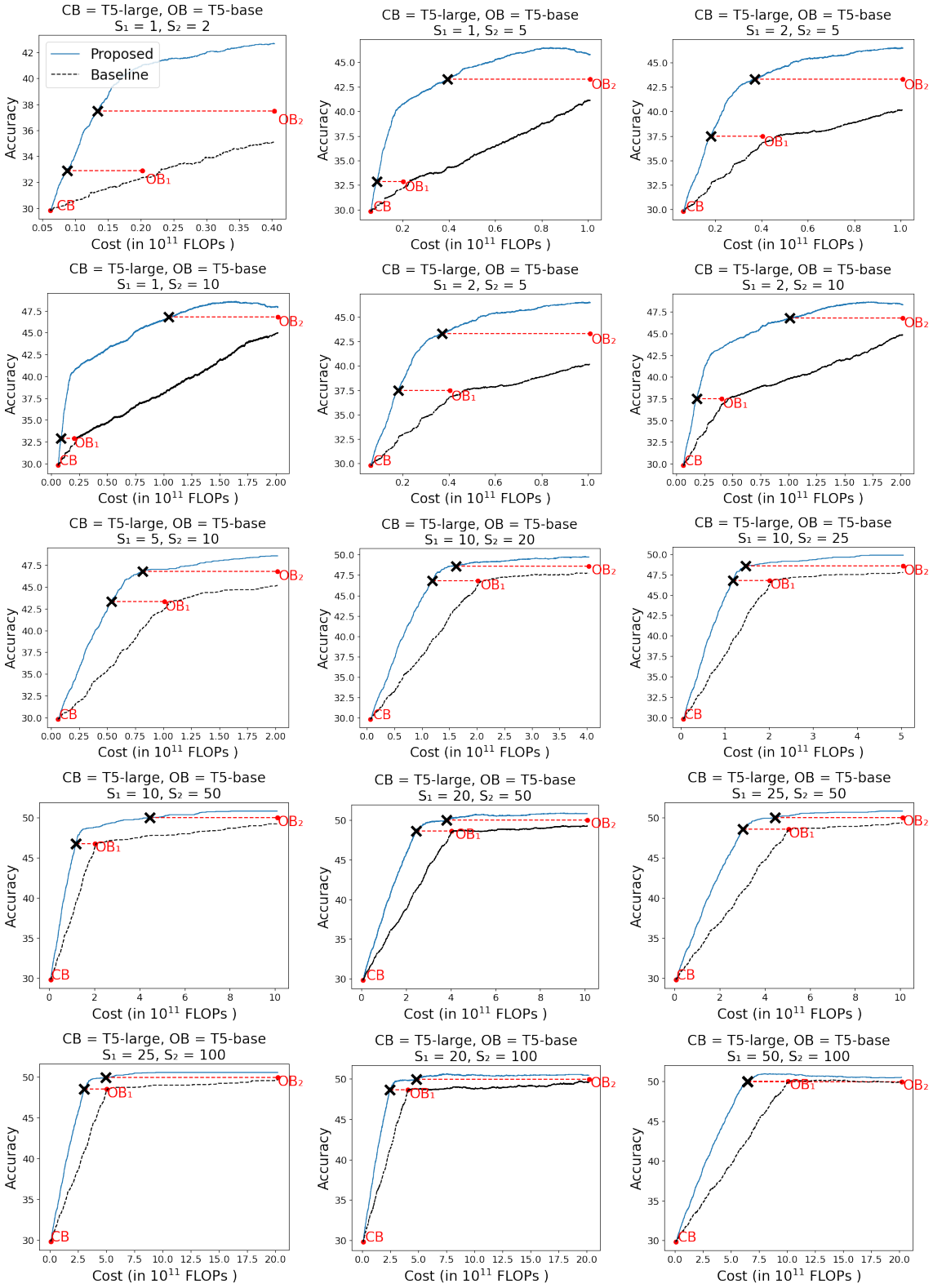


Figure 7: Accuracy-cost curves of the proposed cascading system (in blue) and baseline system (in black) for K=2 setting on NQ. Red points correspond to the accuracy and cost values of the individual models $CB$, $OB_1$ (leveraging $S_1$ knowledge statements), and $OB_2$ (leveraging $S_2$ knowledge statements). Points of intersection of red dashed lines drawn from $OB_1$ and $OB_2$ on the blue curve correspond to costs at which the cascading system achieves the same accuracy as $OB_1$ and $OB_2$ respectively. $OB_1$ and $OB_2$ are the same models but differ only in the $S$ value.

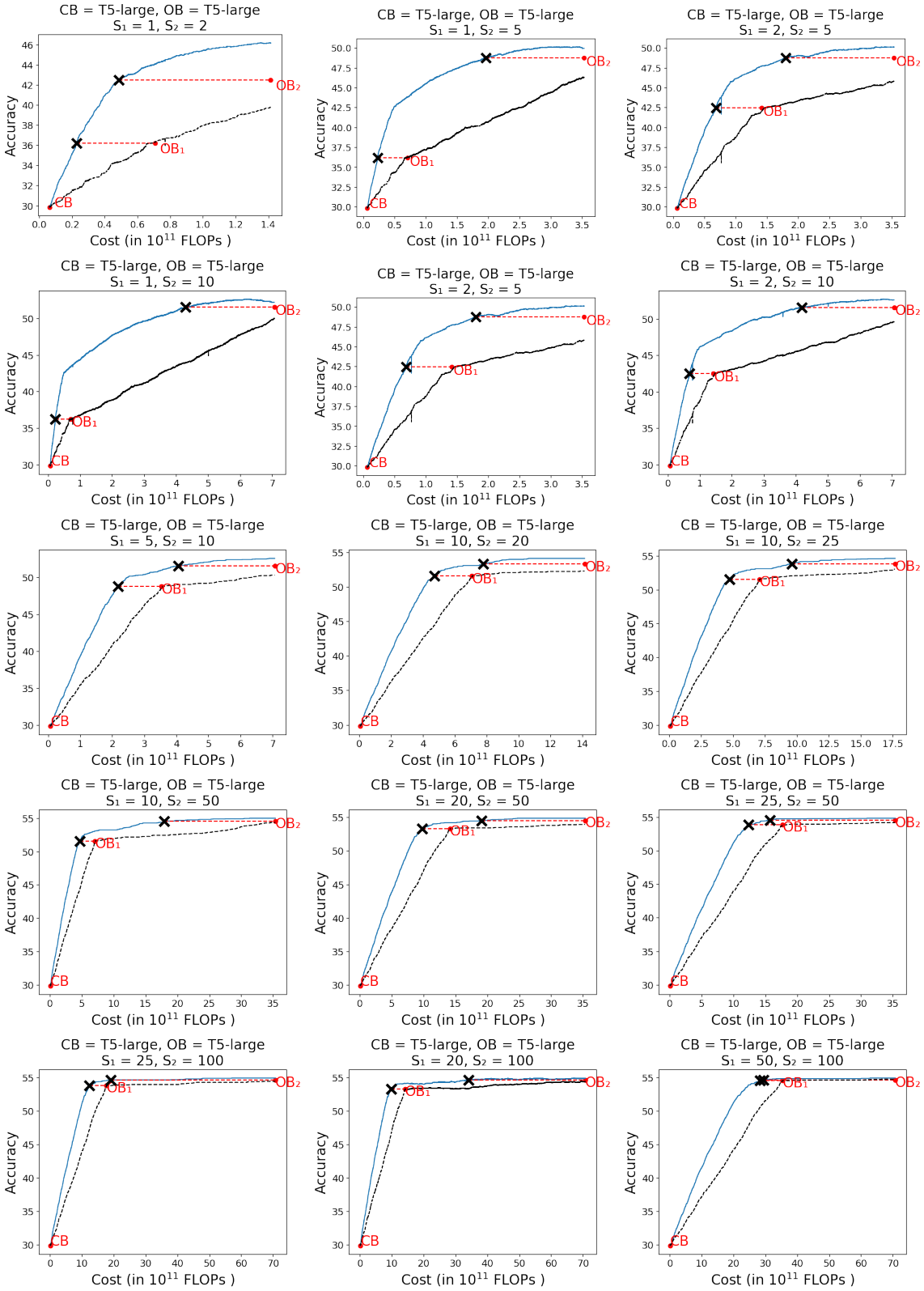


Figure 8: Accuracy-cost curves of the proposed cascading system (in blue) and baseline system (in black) for K=2 setting on NQ. Red points correspond to the accuracy and cost values of the individual models $CB$, $OB_1$ (leveraging $S_1$ knowledge statements), and $OB_2$ (leveraging $S_2$ knowledge statements). Points of intersection of red dashed lines drawn from $OB_1$ and $OB_2$ on the blue curve correspond to costs at which the cascading system achieves the same accuracy as $OB_1$ and $OB_2$ respectively. $OB_1$ and $OB_2$ are the same models but differ only in the $S$ value.