MetaQA: Combining Expert Agents for Multi-Skill Question Answering

Anonymous ACL submission

Abstract

The recent explosion of question answering (QA) datasets and models has increased the interest in the generalization of models across multiple domains and formats by either training on multiple datasets or by combining multiple models. Despite the promising results of multi-dataset models, some domains or QA formats may require specific architectures, and thus the adaptability of these models might be limited. In addition, current approaches for combining models disregard cues such as question-answer compatibility. In this work, we propose to combine expert agents with a novel, flexible, and training-efficient architecture that considers questions, answer predictions, and answer-prediction confidence scores to select the best answer among a list of answer candidates. Through quantitative and qualitative experiments we show that our model i) creates a collaboration between agents that outperforms previous multi-agent and multi-dataset approaches in both in-domain and out-of-domain scenarios, ii) is highly data-efficient to train, and iii) can be adapted to any QA format. We release our code and a dataset of answer predictions from expert agents for 16 QA datasets to foster future developments of multi-agent systems[1].

1 Introduction

The large number of question answering (QA) datasets released in the past years has been accompanied by models specialized on them (Rogers et al., 2021; Dzendzik et al., 2021). These datasets and models differ by domain (e.g., biomedical, Wikipedia, etc), required skills (e.g., numerical, multi-hop, etc), and format (e.g., extractive, multiple-choice, etc). This variety of tasks and overspecialization of the corresponding models have led the community towards developing simple unified models that can generalize across domains and formats through unifying dataset formats (Khashabi et al., 2020), creating models trained on multiple datasets (Fisch et al., 2019; Talmor and Berant, 2019; Khashabi et al., 2020), and designing ensemble methods for QA agents (Geigle et al., 2021). All these research lines have a potential impact on end-user applications because generalization can help create robust systems and ease the implementation of QA models. More abstractly, these research lines also share a central research question: *how to combine QA skills.*

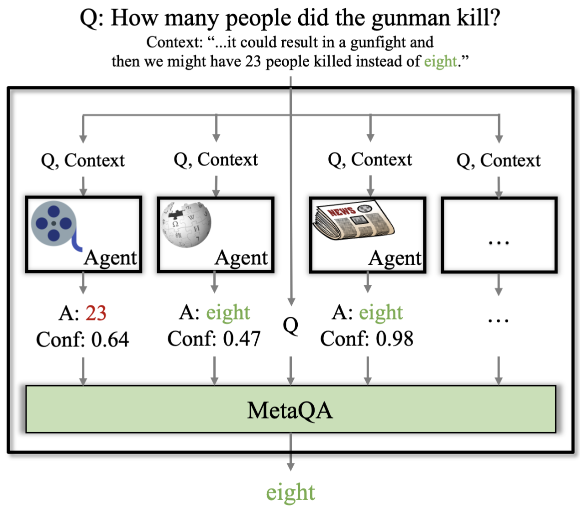

Figure 1: Given a question, each expert agent provides a prediction with a confidence score and MetaQA selects the best answer. Correct answers in green. Wrong answers in red.

We argue that a *one-size-fits-all* architecture may encounter some limitations to combine QA skills. For instance, Raffel et al. (2020) has observed that a single model trained on multiple tasks may underperform the same architecture trained on a single task. An alternative approach is to combine multiple expert agents. Geigle et al. (2021) propose a model that given a question and a list of agents, selects an agent trained on the domain of

[1] https://anonymous.4open.science/r/MetaQA-3468/README.md

the input question. However, even though they achieve a classification accuracy greater than 90%, they disregard the actual predictions and confidence scores from the agents when selecting the output agent, which may result in underestimating high-performing models on out-of-domain questions.

To address the limitations of previous approaches, we propose a novel model to combine heterogeneous expert agents (i.e., different architectures, formats, and tasks). It takes a question, and a list of *candidate answers* with *confidence scores* as input and selects the best answer (Figure 1). We modify the embedding mechanism of the Transformer encoder (Vaswani et al., 2017) to embed the confidence score of each candidate answer. In addition, we use a multi-task training objective that makes the model learn two complementary tasks: *selecting the best candidate answer* and *identifying agents trained on the domain of the input question*.

Unlike multi-dataset models, our approach learns to match questions with answers, an immensely easier task than end-to-end QA itself. This makes our model remarkably data efficient as it only needs 16% of the amount of data needed to train multi-dataset models.

We compile a list of 16 QA datasets that encompass different domains, formats, and reasoning skills to conduct experiments on. Through quantitative experiments we show that our MetaQA i) establishes a successful collaboration between agents, ii) outperforms multi-agent and multi-dataset models in both in-domain and out-of-domain scenarios, iii) excels in minority domains, and iv) is highly efficient to train. Our contributions are:

- A new approach for multi-skill QA that establishes a collaboration between agents.
- A model called MetaQA that utilizes question, answer, and confidence scores to select the best candidate answer for a given question.
- Extensive analyses showing the successful collaboration between agents and the training efficiency of our approach.
- A dataset of (*QA Agents*, *Questions*, and *answer predictions*) triples that cover different QA formats, domains, and skills to foster future developments of multi-agent models.

2 Related Work

Currently, there are two approaches to solve questions from multiple QA domains: ensemble models and multi-dataset models. The former combines multiple QA agents trained on a single dataset and the latter is a model trained on multiple datasets.

Ensemble Methods for QA. A well-known method for combining expert agents is the Mixture of Experts (MoE). It requires training a set of models and combining their outputs with a gating mechanism (Jacobs et al., 1991). However, this approach would require jointly training multiple agents, which can be extremely expensive, and sharing a common output space to combine the agents. These limitations make it unfeasible to implement in our setup, where a large number of heterogeneous agents are combined (i.e., agents with different architectures, target tasks, and output formats such as integers for multiple-choice or answer spans for span extraction).

Recently, Geigle et al. (2021) proposed agent classifiers on top of a Transformer to identify the most appropriate agent for a given question. However, they disregard answer predictions when selecting the agent and hence, agents that are effective in out-of-domain questions are underestimated. Lastly, Friedman et al. (2021) average the weights of adapters (Houlsby et al., 2019) trained on single datasets to obtain a multi-dataset model. However, their architecture is limited to span extraction.

Multi-dataset models consist of training a model on various datasets to generalize it to multiple domains. Talmor and Berant (2019) conduct extensive analysis of the generalization of QA models using ten datasets. However, they only experiment on extractive tasks and, due to their model architecture (BERT for span extraction), it is not possible to extend it to other tasks such as abstractive or visual QA. Fisch et al. (2019) created a competition on QA generalization using 18 datasets. These datasets are from very different domains such as Wikipedia and biomedicine, among others. However, they also focus only on extractive datasets.

Lastly, Khashabi et al. (2020) takes one step further showing that the different QA formats can complement each other to achieve a better generalization. They use an encoder-decoder architecture and transform the questions into a common format. However, we argue that their approach is limited by the fact that some questions may require a specific skill that must be modeled in a particular manner (e.g. numerical reasoning) and, this is not possible with their simple encoder-decoder.

3 Model

We propose a new model, shown in Figure 2, to combine QA agents by integrating cues of the QA task, such as question-answer compatibility. We also define two complementary tasks: i) in-domain agent selection (Agent Selection Networks, AgSeN, in Figure 2) and ii) answer selection (AnsSel network in Figure 2). The division of the problem into these two learnable tasks is vital to ensure that MetaQA considers out-of-domain agents that can give a correct answer, unlike TWEAC (Geigle et al., 2021). To achieve this, the backbone of our architecture relies on an encoder Transformer (Vaswani et al., 2017) whose input is the concatenation of the question with the candidate answers from each agent. Each answer is separated by a new token `[ANS]` that informs the model of the beginning of a new answer candidate.

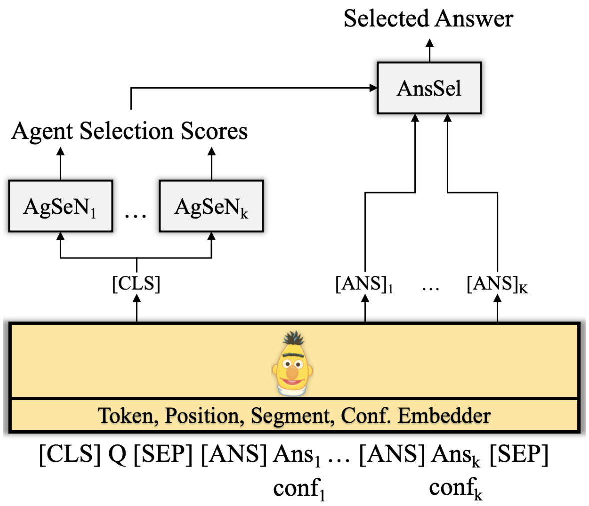

Figure 2: MetaQA architecture. The Agent Selection Networks, AgSeN, identifies the best agent for the input question Q and the Answer Selection, AnsSel, selects the best answer prediction. conf_k is the confidence score from the agent for answer k.

We devise a new embedding for the Transformer encoder to include the confidence score of the predictions of each agent (Figure 3). While the original encoder uses the token t_i, position p_i, and segment s_i embeddings, we add an agent confidence embedding c_i to these three.

$$x_i = t_i + p_i + s_i + c_i \tag{1}$$

As usual, the segment embedding, s_i is used to distinguish two parts of the input: the input question (segment A) and the candidate answers (segment B). As for the new c_i, it is obtained with a feed-forward network f that takes an answer confidence conf_i and creates an embedding c_i.

$$c_i = \begin{cases} f(\text{conf}_j), & \text{if } i \in Idx([\text{ANS}]\ \text{Ans}_j) \\ f(0), & \text{otherwise} \end{cases} \tag{2}$$

where Idx is a function that given a list of tokens returns their indexes in the encoder input.

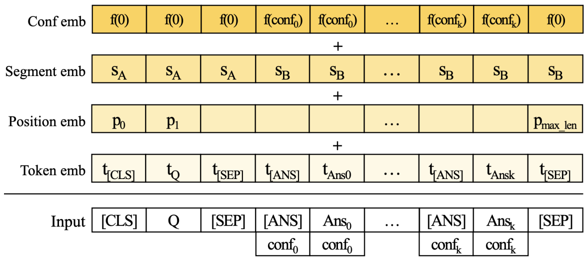

Figure 3: Description of our novel embedding system including confidence scores from the agents.

We leverage two types of embeddings from the output of the encoder. The first one is the embedding of the `[CLS]` token. This embedding captures information about the domain of the input question and is used as the input to k independent feed-forward networks called *Agent Selection Network* (AgSeN) to classify the agent trained on the domain of the input question in the same way as in TWEAC. The second type of embedding used is the embedding of the `[ANS]` tokens. They contain the cues needed to discriminate the best answer to the input question. These `[ANS]` embeddings are concatenated with the model selection scores and input into a final feed-forward network, called *Answer Selection* (AnsSel), that selects the best candidate answer according to the domain of the question and the candidate answers.

3.1 Training

As previously mentioned, our model learns two complementary tasks: i) agent selection and ii) answer selection. Thus, to learn these two tasks we define the following loss function:

$$\ell = \frac{\alpha_1}{k} \sum_{i=0}^{k} \ell_{AgSeN_i} + \alpha_2 \ell_{AnsSel} \tag{3}$$

where ℓ_{AgSeN_i} is the loss of one AgSeN network and ℓ_{AnsSel} the loss of the AnsSel network.

	Dataset	Characteristics
Extractive	SQuAD (Rajpurkar et al., 2016)	Crowdsourced questions on Wikipedia
	NewsQA (Trischler et al., 2017)	Crowdsourced questions about News
	HotpotQA (Yang et al., 2018)	Crowdsourced multi-hop questions on Wikipedia
	SearchQA (Dunn et al., 2017)	Web Snippets, Trivia questions from J! Archive
	NQ (Kwiatkowski et al., 2019)	Wikipedia, real user queries on Google Search
	TriviaQA-web (Joshi et al., 2017)	Web Snippets, crowdsorced trivia questions
	QAMR (Michael et al., 2018)	Wikipedia, predicate-argument understanding
	DuoRC (Saha et al., 2018)	Movie Plots from IMDb and Wikipedia
Multiple-Choice	RACE (Lai et al., 2017)	Exams requiring passage summarization and attitude analysis
	CSQA (Talmor et al., 2019)	Web Snippets, common-sense reasoning
	BoolQ (Clark et al., 2019)	Wikipedia, Yes/No questions
	HellaSWAG (Zellers et al., 2019)	Completing sentences using common sense
	SIQA (Sap et al., 2019)	Common sense in social interactions
Abs.	DROP (Dua et al., 2019)	Wikipedia, numerical reasoning
	NarrativeQA (Kočiský et al., 2018)	Books, Movie Scripts
MM	HybridQA (Chen et al., 2020)	Wikipedia tables and paragraphs

Table 1: Summary of the datasets used. Abs. stands for abstractive and MM for multi-modal.

#	Expert Agents	Used for
1	Span-BERT Large (Joshi et al., 2020) for SQuAD	all extractive + DROP
2	Span-BERT Large for NewsQA	all extractive + DROP
3	Span-BERT Large for HotpotQA	all extractive + DROP
4	Span-BERT Large for SearchQA	all extractive + DROP
5	Span-BERT Large for NQ	all extractive + DROP
6	Span-BERT Large for TriviaQA-web	all extractive + DROP
7	Span-BERT Large for QAMR	all extractive + DROP
8	Span-BERT Large for DuoRC	all extractive + DROP
9	RoBERTa Large (Liu et al., 2019) for RACE	all multiple choice
10	RoBERTa Large for HellaSWAG	all multiple choice
11	RoBERTa Large for SIQA	all multiple choice
12	AlBERT xxlarge-v2 (Lan et al., 2020) for CSQA	all multiple choice
13	BERT Large-wwm (Devlin et al., 2019) for BoolQ	BoolQ
14	TASE (Segal et al., 2020) for DROP	DROP
15	Adapter BART Large (Pfeiffer et al., 2020) for NarrativeQA	NarrativeQA
16	Hybrider (Chen et al., 2020) for HybridQA	HybridQA

Table 2: List of the expert agents and datasets in which they are used.

We compute the loss of the AnsSel network using Cross-Entropy while for the AgSeN networks we use the Binary Cross Entropy.

The labels to train AnsSel are obtained by comparing the prediction of each agent with the correct answer. If the F1 score is higher than a threshold, θ, we consider the prediction as correct. As for AgSeN$_i$, its training label is 1 when the input question is from the training set of the i^{th} agent.

4 Experimental Setup

4.1 Datasets

We have collected a series of QA datasets covering different formats, domains, and reasoning skills (Table 1). In particular, we use four formats: extractive, multiple-choice, abstractive, and multimodal.

For extractive, we use the MRQA 2019 shared task collection (Fisch et al., 2019), QAMR (Michael et al., 2018), and DuoRC (Saha et al., 2018). We add these two additional datasets to add more diversity to the training set. In detail, QAMR requires predicate-argument understanding, a skill that agents should have to solve most QA datasets. As for DuoRC, it is the only dataset in our collection on the film domain, and this allows us to study transfer learning from other domains. The multiple-choice datasets require boolean reasoning, commonsense, and passage summarization skills and as we can observe in Table 1, there is an overlap in the reasoning skills required to solve these datasets. Lastly, we include abstractive QA following (Khashabi et al., 2020) and multimodal datasets to show that our approach can solve any type of question while multi-dataset models are limited to certain formats.

Most of these datasets do not have the labels of the test set publicly available, except for RACE and NarrativeQA. Since we need to do hyperparameter tuning and hypothesis testing to compare models, we divide the public dev set into an in-house dev set and test sets following (Joshi et al., 2020). In this way, we conduct hyperparameter tuning on the dev set and hypothesis testing on the test set.

4.2 Expert Agents

To guarantee a fair comparison with MultiQA, we have trained all the agents for extractive datasets using the same architecture as MultiQA, span-BERT, a BERT model pretrained for span extraction tasks that clearly outperforms BERT on the MRQA 2019 shared task (Joshi et al., 2020). More details on the implementation are provided in Appendix A.2.

For the remaining datasets, we use agents that are publicly available on HuggingFace or Github

Dataset	MetaQA	TWEAC	Exp. Agent	UnifiedQA	MultiQA
SQuAD	91.98±0.11†	89.09±0.36	92.92	90.81	**93.14±0.18**
NewsQA	71.71±0.21†	66.86±0.75	**73.68**	65.57	73.59±0.60
HotpotQA	79.27±0.15†	74.96±0.59	80.60	77.92	**81.68±0.22**
SearchQA	**81.98±0.25**†‡	80.41±0.22	81.04	81.61	80.45±1.82
TriviaQA-web	**80.63±0.26**†‡	76.55±0.15	79.34	72.34	77.76±4.15
NQ	81.20±0.18†	78.06±0.37	81.97	75.58	**82.57±0.30**
DuoRC	**51.24±0.20**†‡	44.28±0.23	43.77	34.65	46.99±0.15
QAMR	83.78±0.14†	78.77±0.48	84.00	82.70	**84.62±0.14**
BoolQ	73.14±0.23†	72.20±0.03	72.17	**81.34**	n.a.
CSQA	**78.66±0.19**†	77.18±0.18	78.56	58.43	n.a.
HellaSWAG	73.19±1.01	77.12±0.30	**77.14**	36.01	n.a.
RACE	84.71±0.05†	83.02±0.27	**84.78**	69.65	n.a.
SIQA	74.17±0.64	75.39±0.05	**75.44**	61.62	n.a.
DROP	73.04±1.98	**74.61±0.00**	**74.61**	42.45	n.a.
NarrativeQA	**67.19±0.00**	**67.19±0.00**	**67.19**	57.82	n.a.
HybridQA	**50.94±0.00**	**50.94±0.00**	**50.94**	n.a	n.a

Table 3: MetaQA (ours) and the baselines on the test set of each dataset. Best results in bold. † represents that MetaQA is statistically significant better than TWEAC. ‡ represents that MetaQA is statistically significant better than MultiQA. n.a means that the system cannot model the dataset.

with a performance close to the current state of the art. A summary of the agents is provided in Table 2 and links to download them in Appendix A.1.

4.3 Baselines

We compare our approach with three types of models: i) multi-agent systems, ii) multi-dataset models, and iii) expert agents. The first family is represented by our main baseline, TWEAC, a model that maps questions to agents that can solve them (Geigle et al., 2021). Our MetaQA also ascribes to this family. As for the second family of models, we use the currently most representative works, MultiQA (Talmor and Berant, 2019) and UnifiedQA (Khashabi et al., 2020). MultiQA is a transformer encoder with a span-extraction layer trained on multiple extractive QA datasets. Because of this span-extraction layer, it can only solve extractive QA tasks. UnifiedQA, on the other hand, can solve any QA task that can be converted into text-to-text thanks to its architecture, an encoder-decoder transformer (i.e., extractive, abstractive, and multiple-choice). Lastly, we also compare our proposal with expert agents in each dataset, i.e., models trained on a single dataset.

4.4 Evaluation

We evaluate our model and the baselines using the official metrics of each dataset, i.e., macro-average F1 for extractive, accuracy for multiple-choice, and rouge-L for abstractive. In the particular case of DROP, the official metric is macro-average F1, and thus, we also use it. The reported results are the means and standard deviations of the models trained with five different seeds except for UnifiedQA, which would be too expensive to compute. We use a two-tailed T-Test to compare the models with a p-value of 0.05.

5 Results

In the experiments, we answer the following questions: i) is MetaQA able to combine multiple agents without undermining the performance of each one (§5.1), ii) is it robust on out-of-domain scenarios? (§5.2), iii) how does agent collaboration work? (§5.3), iv) how data-efficient is MetaQA? (§5.4), and v) what is the effect of each module of MetaQA? (§5.5).

5.1 Overall Performance

In Table 3, we compare the performance of MetaQA with the baselines and prior works. To begin with, our proposal outperforms TWEAC in

Dataset	NewsQA	HotpotQA	SearchQA	TriviaQA	NQ	DuoRC	QAMR	CSQA	HellaSWAG	SIQA	DROP
MetaQA	71.46	79.37	81.87	80.65	81.08	51.01	83.87	78.40	72.14	73.90	74.96
UnifiedQA	65.57	77.92	81.61	72.34	75.58	34.65	82.70	58.43	36.01	61.62	42.45
OOD MetaQA	**64.39**	**70.62**	**67.82**	**77.76**	**65.52**	**51.23**	**71.90**	46.48	**55.09**	**59.77**	**22.36**
OOD UnifiedQA	60.12	62.21	63.02	69.33	61.49	32.84	70.07	**50.57**	29.35	44.93	22.30

Table 4: Results of leave-one-out ablation. Out-of-domain (OOD) models are trained on all the datasets except the target dataset. Best OOD results in bold. Underlined results reflect OOD MetaQA outperforming full UnifiedQA.

all datasets except HellaSWAG and SIQA. On average, MetaQA achieves an average performance boost of 1.8 with respect to TWEAC, and more importantly, the performance boost is greater than 4 points on HotpotQA, DuoRC, NewsQA, QAMR, and TriviaQA. Particularly, there is an astonishing 6.8 points performance boost on DuoRC. This is achieved thanks to the collaboration between the agents established by MetaQA. In more detail, while TWEAC only attempts to predict the agent trained on the domain of the input question, we aim to retrieve the best answer prediction, even if it comes from a model trained on a completely different dataset. For instance, in DuoRC, our MetaQA selects the in-domain agent only for 43% of its questions, i.e, most of the questions are assigned to agents that are not trained on DuoRC.

When comparing to UnifiedQA, we can observe the limitations of its architecture. For example, the performance in DROP is clearly far from our MetaQA. The reason for this is that while the expert agent used by MetaQA is designed for numerical reasoning, UnifiedQA does not have any mechanism to achieve this, and since it is designed as a general model for text-to-text generation, it cannot be augmented with special reasoning modules. The same phenomenon occurs in the multiple-choice datasets and in some minority domains in extractive QA (i.e., NewsQA and DuoRC). The only exception is in BoolQ, where UnifiedQA achieves the best results. However, this is because T5 (Raffel et al., 2020), on which UnifiedQA is trained, is already one of the SOTA models, while the agent we use has lower performance and was the only publicly available model in HuggingFace's Model Hub at the time of experimentation.

Lastly, compared to our model, MultiQA achieves an average 0.24 performance increase. However, our model was trained on only 13% of its training set as later discussed in §5.4. In addition, our proposed approach achieves a striking 4.15 points performance boost on DuoRC, a 2.73 on TriviaQA-web, and a 1.55 on SearchQA thanks to the collaboration between the agents. We also observe that MultiQA mostly outperforms the expert agents on the Wikipedia-based datasets (i.e., SQuAD, HotpotQA, NQ, and QAMR). This suggests that MultiQA benefits from the additional Wikipedia data but struggles with other minority domains. On the other hand, our approach excels on those minority domains (i.e., SearchQA, TriviaQA-web, and DuoRC) outperforming MetaQA by an average of 2.88. This shows the successful collaboration between the agents and MetaQA's ability to adapt to new domains.

5.2 Leave-One-Out Ablation

In this experiment, we analyze whether the combination of expert agents can successfully solve an out-of-domain dataset. We conduct a leave-one-out ablation test in both MetaQA and UnifiedQA. In the case of MetaQA, it is possible to *switch-off* agents without retraining the model. We just need to set to null all the predictions of the ablated agent. On the other hand, in UnifiedQA we have to retrain the model without the target dataset for each dataset. Table 4 shows that the out-of-domain MetaQA outperforms UnifiedQA in all datasets except in CommonSenseQA by an average of 9.14 points. In addition, in three datasets (DuoRC, HellaSWAG, and TriviaQA-web), the ablated MetaQA even outperforms the full UnifiedQA trained on those datasets. This is another piece of evidence of the successful collaboration between agents and suggests that agent collaboration might be more suitable than transfer learning in certain situations.

5.3 MetaQA Analysis

We further analyze the behavior of our proposed model by inspecting its predictions. In particular, we investigate the collaboration between the agents for DuoRC, SearchQA, and TriviaQA, where this collaboration is particularly strong.

In DuoRC, the most helpful out-of-domain (ood) agent is NewsQA with a chosen rate of 18.2% in the test set. This might be due to the question

Dataset	Question	In-domain Agent	OOD Agent
DuoRC	Who does Rocky Balboa work for as an enforcer?	Adrian	Tony Gazzo (NewsQA Agent)
TriviaQA-web	Who played the character Mr Chips in the 2002 TV adaptation of Goodbye Mr Chips	Timothy Carroll	MartinClunes (DuoRC Agent)
SearchQA	This short story, written around 1820, contains the line "If I can but reach that bridge... I am safe"	Legend	Legend of Sleepy Hollow (TriviaQA Agent)

Table 5: Examples of questions where our MetaQA system disregard the in-domain agent due to their incorrect predictions (in red) and selects and an out-of-domain (OOD) agent that returns the right answer (in green).

types of DuoRC and NewsQA. DuoRC's questions are crowdsourced and are predominately *who-questions* (42% of the training set as shown in Appendix 11). NewsQA's questions are also crowdsourced and have a high proportion of *who-questions* (24%). The other datasets with a high amount of *who-questions* are NQ and SearchQA. However, the questions of these two are very different in style to DuoRC (i.e., real user queries and trivia from a TV show). An example of this DuoRC-NewsQA agents collaboration is shown in the first row of Table 5.

In TriviaQA-web, the second most commonly used agent is trained on DuoRC. We randomly sampled 50 QA pairs where DuoRC is the selected agent and returns the right answer. In 20% of the cases, the question was about a movie or book plot, which indicates that our MetaQA successfully recognizes that this ood agent is able to respond to this type of question. An example of this collaboration is shown in the second row in Table 5.

In SearchQA, the most helpful ood agent is TriviaQA (5% chosen rate). This might be due to their similarities (Table 1). Within the pool of instances where the in-domain agent fails and the TriviaQA agent provides the right answer, we randomly analyzed 50 instances and discovered that in 84% of the cases, the in-domain agent returns a partially correct answer (i.e., it fails to identify the exact answer boundaries), and in those cases, the ood agent was able to identify the correct answer boundaries. This is another example of the successful agent collaboration achieved by our MetaQA. Even though the in-domain agent almost have the correct answer, MetaQA selects an ood agent that gives a better answer as shown in the last row on Table 5.

The main limitation of our approach is that when no agent has a correct answer, MetaQA would return an incorrect answer. Table 6 describes how often this scenario occurs. In extractive datasets, without the outliers (i.e., SQuAD and DuoRC), we observe this to be 18% on average per dataset. This percentage drops to 8.35% in the multiple choice datasets (without BoolQ, another outlier). As for NarrativeQA and HybridQA, since we only use one agent for each of them and these agents have a relatively low performance, there is a large number of unsolvable questions.

Dataset	% Unsolvable
SQuAD	3.92
NewsQA	26.88
HotpotQA	19.93
SearchQA	13.97
NQ	19.15
TriviaQA-web	12.25
QAMR	15.81
DuoRC	47.41
BoolQ	1.47
SIQA	8.90
HellaSWAG	8.90
CSQA	9.00
RACE	6.61
DROP	21.77
NarrativeQA	55.71
HybridQA	56.09

Table 6: Percentage of unsolvable questions for our MetaQA with the selected agents, i.e., none of the agents can give a correct answer.

5.4 Efficiency of MetaQA

We trained MetaQA with bins of QA instances for each dataset and observe that the training converges with only 10K instances/per dataset (i.e., 160K instances including all datasets). This is only 16%

of the data needed to train UnifiedQA (900K instances excluding HybridQA) and 13% of the data needed to train MetaQA (600K of extractive QA instances). The reason for this large saving is that MetaQA only has to learn how to match questions with answers because it reuses publicly available agents. On the other hand, multi-dataset models need to learn how to solve questions (i.e., language understanding, reasoning skills, etc), a much more complex task.

As for inference time, if all the agents fit on memory[2], then multi-datasets models and our MetaQA would have comparable running times. For example, compared to MultiQA, since our extractive agents use the same architecture as MultiQA, running the agents would take the same amount of time as running MultiQA. Then, we would need to select the answer. However, our MetaQA only takes 0.05s/question to select the best candidate answer. This makes it fast enough to not be noticeable by the users. On the other hand, if the agents do not fit in memory at the same time, it would be needed to run them sequentially. Yet, this might not be a problem because it is possible to predict in advance which agents are more likely to give a correct answer to a given question (Geigle et al., 2021; Garg and Moschitti, 2021), which we leave as future work. This would allow us to skip some agents at run-time and improve the running time dramatically in low-memory scenarios.

5.5 Ablation Study

Lastly, we quantitatively measure the impact of each feature of MetaQA on its overall performance. The first row of Table 7 shows that removing the loss of the Agent Selection Network (AgSeN) hurts the performance of MetaQA. This manifests that our intuition of considering in-domain agents without falling into the *argumentum ad verecundiam* fallacy is correct. Lastly, the second row shows that the confidence embeddings provide key information to MetaQA to select an answer. For instance, an in-domain agent could have a prediction with low confidence because it does not know the answer while an out-of-domain agent could have the correct answer and be certain about it.

[2]In our hardware and with our experimental setup, all agents and MetaQA fit on our GPU memory.

Model	Avg. Downgrade
Full model	-
$-\ell_{AgSeN}$	-0.45
$-$ Conf. Emb.	-0.46

Table 7: Average performance loss across all datasets of each ablated model compared to the full model.

6 Conclusion

In this work, we propose a new system to combine expert agents for question answering (QA) called MetaQA. It considers questions, answer predictions, and confidence scores from the agents to select the best answer to a question. Through quantitative experiments, we show that our model avoids the limitations of multi-dataset models and outperforms the baselines in both in-domain and out-of-domain scenarios thanks to the agent collaboration established by MetaQA. Additionally, since MetaQA learns how to match questions with answers instead of end-to-end QA, it is higly data-efficient to train.

We leave as future work: i) combining partially correct answer predictions to generate a better answer, ii) adding new agents without retraining the whole MetaQA by fixing most of the weights and only training the weights of the Agent Selection Network, and iii) identifying *a priori* agents that are likely to give an incorrect answer to skip them at run-time.

Ethics Discussion

The proposed model, MetaQA, cannot generate unfair, biased, or harmful contents given that the expert agents it aggregates are fair because MetaQA does not generate content, rather it selects from Expert Agents. Future work should address how to identify unfair content to avoid selecting it. Similarly, the veracity of the answers given by MetaQA rely on the expert agents and the evidence documents used.

References

Wenhu Chen, Hanwen Zha, Zhiyu Chen, Wenhan Xiong, Hong Wang, and William Yang Wang. 2020. HybridQA: A dataset of multi-hop question answering over tabular and textual data. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1026–1036, Online. Association for Computational Linguistics.

Christopher Clark, Kenton Lee, Ming-Wei Chang, Tom Kwiatkowski, Michael Collins, and Kristina Toutanova. 2019. BoolQ: Exploring the surprising difficulty of natural yes/no questions. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2924–2936, Minneapolis, Minnesota. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota. Association for Computational Linguistics.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 2368–2378, Minneapolis, Minnesota. Association for Computational Linguistics.

Matthew Dunn, Levent Sagun, Mike Higgins, V Ugur Guney, Volkan Cirik, and Kyunghyun Cho. 2017. Searchqa: A new q&a dataset augmented with context from a search engine. *arXiv preprint arXiv:1704.05179*.

Daria Dzendzik, Jennifer Foster, and Carl Vogel. 2021. English machine reading comprehension datasets: A survey. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 8784–8804, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. 2019. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China. Association for Computational Linguistics.

Dan Friedman, Ben Dodge, and Danqi Chen. 2021. Single-dataset experts for multi-dataset question answering. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 6128–6137, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Siddhant Garg and Alessandro Moschitti. 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*, pages 7329–7346, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Gregor Geigle, Nils Reimers, Andreas Rücklé, and Iryna Gurevych. 2021. Tweac: Transformer with extendable qa agent classifiers. *arXiv preprint*, abs/2104.07081.

Neil Houlsby, Andrei Giurgiu, Stanislaw Jastrzebski, Bruna Morrone, Quentin De Laroussilhe, Andrea Gesmundo, Mona Attariyan, and Sylvain Gelly. 2019. Parameter-efficient transfer learning for nlp. In *International Conference on Machine Learning*, pages 2790–2799. PMLR.

Robert A Jacobs, Michael I Jordan, Steven J Nowlan, and Geoffrey E Hinton. 1991. Adaptive mixtures of local experts. *Neural Computation*, 3(1):79–87.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. 2020. SpanBERT: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics*, 8:64–77.

Mandar Joshi, Eunsol Choi, Daniel Weld, and Luke Zettlemoyer. 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)*, pages 1601–1611, Vancouver, Canada. Association for Computational Linguistics.

Daniel Khashabi, Sewon Min, Tushar Khot, Ashish Sabharwal, Oyvind Tafjord, Peter Clark, and Hannaneh Hajishirzi. 2020. UNIFIEDQA: Crossing format boundaries with a single QA system. In *Findings of the Association for Computational Linguistics: EMNLP 2020*, pages 1896–1907, Online. Association for Computational Linguistics.

Tomáš Kočiský, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gábor Melis, and Edward Grefenstette. 2018. The NarrativeQA reading comprehension challenge. *Transactions of the Association for Computational Linguistics*, 6:317–328.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Guokun Lai, Qizhe Xie, Hanxiao Liu, Yiming Yang, and Eduard Hovy. 2017. RACE: Large-scale ReAding comprehension dataset from examinations. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 785–794, Copenhagen, Denmark. Association for Computational Linguistics.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. Albert: A lite bert for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Quentin Lhoest, Albert Villanova del Moral, Yacine Jernite, Abhishek Thakur, Patrick von Platen, Suraj Patil, Julien Chaumond, Mariama Drame, Julien Plu, Lewis Tunstall, Joe Davison, Mario Šaško, Gunjan Chhablani, Bhavitvya Malik, Simon Brandeis, Teven Le Scao, Victor Sanh, Canwen Xu, Nicolas Patry, Angelina McMillan-Major, Philipp Schmid, Sylvain Gugger, Clément Delangue, Théo Matussière, Lysandre Debut, Stas Bekman, Pierric Cistac, Thibault Goehringer, Victor Mustar, François Lagunas, Alexander Rush, and Thomas Wolf. 2021. Datasets: A community library for natural language processing. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 175–184, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

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

Julian Michael, Gabriel Stanovsky, Luheng He, Ido Dagan, and Luke Zettlemoyer. 2018. Crowdsourcing question-answer meaning representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 2 (Short Papers)*, pages 560–568, New Orleans, Louisiana. Association for Computational Linguistics.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Kopf, Edward Yang, Zachary DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. 2019. Pytorch: An imperative style, high-performance deep learning library. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems 32*, pages 8024–8035. Curran Associates, Inc.

Jonas Pfeiffer, Andreas Rücklé, Clifton Poth, Aishwarya Kamath, Ivan Vulić, Sebastian Ruder, Kyunghyun Cho, and Iryna Gurevych. 2020. AdapterHub: A framework for adapting transformers. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 46–54, Online. Association for Computational Linguistics.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the limits of transfer learning with a unified text-to-text transformer. *Journal of Machine Learning Research*, 21(140):1–67.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2383–2392, Austin, Texas. Association for Computational Linguistics.

Anna Rogers, Matt Gardner, and Isabelle Augenstein. 2021. Qa dataset explosion: A taxonomy of nlp resources for question answering and reading comprehension. *arXiv preprint arXiv:2107.12708*.

Amrita Saha, Rahul Aralikatte, Mitesh M. Khapra, and Karthik Sankaranarayanan. 2018. DuoRC: Towards complex language understanding with paraphrased reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1683–1693, Melbourne, Australia. Association for Computational Linguistics.

Maarten Sap, Hannah Rashkin, Derek Chen, Ronan Le Bras, and Yejin Choi. 2019. Social IQa: Commonsense reasoning about social interactions. 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)*, pages 4463–4473, Hong Kong, China. Association for Computational Linguistics.

Elad Segal, Avia Efrat, Mor Shoham, Amir Globerson, and Jonathan Berant. 2020. A simple and effective model for answering multi-span questions. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 3074–3080, Online. Association for Computational Linguistics.

Alon Talmor and Jonathan Berant. 2019. MultiQA: An empirical investigation of generalization and transfer in reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4911–4921, Florence, Italy. Association for Computational Linguistics.

Alon Talmor, Jonathan Herzig, Nicholas Lourie, and Jonathan Berant. 2019. CommonsenseQA: A question answering challenge targeting commonsense knowledge. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4149–4158, Minneapolis, Minnesota. Association for Computational Linguistics.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. NewsQA: A machine comprehension dataset. In *Proceedings of the 2nd Workshop on Representation Learning for NLP*, pages 191–200, Vancouver, Canada. Association for Computational Linguistics.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Remi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander Rush. 2020. Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Rowan Zellers, Ari Holtzman, Yonatan Bisk, Ali Farhadi, and Yejin Choi. 2019. HellaSwag: Can a machine really finish your sentence? In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4791–4800, Florence, Italy. Association for Computational Linguistics.

A Appendix

A.1 Expert Agents

#	Expert Agents	Link
1	Span-BERT Large (Joshi et al., 2020) for SQuAD	in-house trained
2	Span-BERT Large for NewsQA	in-house trained
3	Span-BERT Large for HotpotQA	in-house trained
4	Span-BERT Large for SearchQA	in-house trained
5	Span-BERT Large for NQ	in-house trained
6	Span-BERT Large for TriviaQA-web	in-house trained
7	Span-BERT Large for QAMR	in-house trained
8	Span-BERT Large for DuoRC	in-house trained
9	RoBERTa Large (Liu et al., 2019) for RACE	https://huggingface.co/LIAMF-USP/roberta-large-finetuned-race
10	RoBERTa Large for HellaSWAG	https://huggingface.co/prajjwal1/-roberta_hellaswag
11	RoBERTa Large for SIQA	in-house trained
12	AlBERT xxlarge-v2 (Lan et al., 2020) for CSQA	https://huggingface.co/danlou/-albert-xxlarge-v2-finetuned-csqa
13	BERT Large-wwm (Devlin et al., 2019) for BoolQ	https://huggingface.co/lewtun/-bert-large-uncased-wwm-finetuned-boolq
14	TASE (Segal et al., 2020) for DROP	https://github.com/eladsegal/-tag-based-multi-span-extraction
15	Adapter BART Large (Pfeiffer et al., 2020) for NarrativeQA	in-house trained
16	Hybrider (Chen et al., 2020) for HybridQA	https://github.com/wenhuchen/-HybridQA

Table 8: List of the expert agents and datasets in which they are used for.

Table 8 provides the links to download the expert agents used in this work.

A.2 Implementation

Our model was implemented using PyTorch (Paszke et al., 2019) and HuggingFace's Transformers library (Wolf et al., 2020) with an Nvidia A100 and 16GB RAM. Both MetaQA and MultiQA were implemented using Span-BERT large (335M parameters) while UnifiedQA uses T5-base (220M parameters, the closest to the 335M of our MetaQA). The score embedder for MetaQA is implemented as a linear layer with an input size of 1 and output size of 1024 (i.e., the hidden size of Span-BERT Large). α_1 and α_2 in Eq. 3 are set to 0.5 and 1 respectively. The Agent Selection Networks are implemented as a linear layer with an input size of 1024 and an output size of 1. Lastly, the Answer Selection Network (AnsSel) is also implemented as a linear layer with an input size of *number-of-agents* $\times$ 1025 (Span-BERT's hidden size + 1 from

Dataset	Train	Validation	Test
SQuAD	86573	5253	5254
NewsQA	74160	2106	2106
NQ	104071	6418	6418
HotpotQA	72928	2950	2951
TriviaQA-web	61688	3892	3893
SearchQA	117384	8490	8490
DuoRC	58752	13111	13449
QAMR	50615	18908	18770
RACE	87866	4887	4934
CSQA	9741	611	610
HellaSWAG	39905	5021	5021
SIQA	33410	977	977
BoolQ	9427	1635	1635
DROP	77409	4767	4768
NarrativeQA	32747	3461	10557
HybridQA	62682	1733	1733

Table 9: Split sizes of each dataset.

the output of the agent selection network). The threshold θ to determine whether a candidate answer is correct or not to create the labels to train AnsSel is set to 0.7.

MetaQA was trained for one epoch using a batch size of six, a weight decay of 0.01, a learning rate of 5e-5, and 500 warmup steps.

All the extractive agents and MultiQA were trained using the same architecture, Span-BERT large, for two epochs, and with the same hyperparameters: batch size of 16, learning rate of 3e-5, max length of 512, and doc stride of 128.

Lastly, UnifiedQA was trained for two epochs using a batch size of four, a learning rate of 5e-5, a weight decay of 0.01, and was evaluated on the dev set every 100K steps.

Any other parameter used the default value in HuggingFace's Transformers library. Each model was trained five times with different random seeds to do hypothesis testing except for UnifiedQA, which would be too expensive to compute.

We used the implementation of HuggingFace's Dataset library (Lhoest et al., 2021) for the evaluation using EM and F1 metrics, while for the ROGUE metric we used the official implementation[3].

[3]https://pypi.org/project/rouge-score/

A.3 Adding New Agents

Augmenting MetaQA with a new agent only requires adding one more AgSeN network and increasing the output space of the AnsSel network. Thus, it requires retraining the whole architecture (including the Transformer encoder). However, as discussed in §5.4, the training efficiency is one of the strengths of our system.

A.4 Dataset Sizes

Table 9 contains the size of the train, validation, and test splits of each dataset.

A.5 Dataset Licences

Table 10 shows the license of each dataset. In the case of RACE, the authors did not provide any license but specified that it can only be used for non-commercial research purposes. In the case of CommonSenseQA and SIQA there is no license available, but they are freely available to download. Our use of these datasets comply with their licenses and intended uses.

Dataset	**License**
MRQA	MIT
DuoRC	MIT
QAMR	MIT
RACE	NA
CommonSenseQA	NA
HellaSWAG	MIT
SIQA	NA
BoolQ	CC BY-SA 3.0
DROP	CC BY-SA 4.0
NarrativeQA	Apache 2.0
HybridQA	MIT

Table 10: License of each dataset.

A.6 Wh-word Statistics

Table 11 shows the percentage of wh-words per dataset.

Dataset	**what**	**where**	**who**	**when**	**why**	**which**	**how**
SQuAD	56.71	4.55	10.82	7.47	1.48	7.73	11.23
NewsQA	49.52	8.54	24.46	5.01	0.11	3.17	9.19
HotpotQA	37.98	4.61	22.99	2.22	0.05	29.39	2.76
SearchQA	7.55	9.5	32.53	28.66	0.72	18.32	2.72
NQ	16.58	13.05	40.02	20.35	0.63	3.25	6.11
TriviaQA-web	30.16	1.56	15.07	0.72	0.02	50.03	2.44
QAMR	61.75	5.23	17.92	4.59	0.66	3.04	6.82
DuoRC	35.16	9.68	42.32	2.06	2.44	1.89	6.45

Table 11: Statistics of wh-words per dataset.