# 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 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 predictions. 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, 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 research 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 in them (Rogers et al., 2021; Dzendzik et al., 2021). These datasets and models differ by the domain (e.g., biomedical and Wikipedia), required skills (e.g., numerical and multi-hop), and format (e.g., extractive and multiple-choice). 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*.

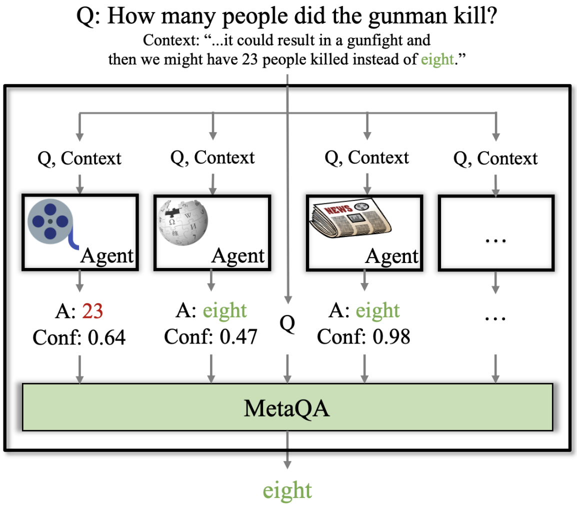


Figure 1: Given a question, each expert agent provides a prediction with a confidence score and MetaQA selects the best answer. Correct answers in green. Wrong answers in red.

We argue that a *one-size-fits-all* architecture may encounter some limitations in combining QA skills. For instance, Raffel et al. (2020) have 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 datasets, selects the dataset from which the question comes. This can be used to identify agents trained on a specific type of questions. However, despite achieving a classification accuracy greater than 90%, this ap-

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

proach underestimates high-performing models on out-of-domain questions.

To address the limitations of previous approaches, we propose a novel model, MetaQA, 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*.

Our approach learns to match questions with answers, an immensely easier task than the end-to-end QA of multi-dataset models. This makes MetaQA remarkably data efficient as it only uses 16% of the training data of multi-dataset models.

We compile a list of 16 QA datasets that encompass different domains, formats, and reasoning skills to conduct experiments. Through quantitative experiments, we show that our MetaQA i) establishes a successful collaboration between agents, ii) outperforms multi-agent and multi-dataset models, 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 for multi-skill QA: multi-agent and multi-dataset models.

**Multi-agent models** consists of combining multiple expert agents. A well-known method is the Mixture of Experts. 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 many 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). Inspired by topic classification, Geigle et al. (2021) proposed mapping questions to QA datasets (topics) to identify agents trained on that type of questions. Although related to us, their work does not attempt to achieve any agent collaboration. Moreover, because of their *topic-classification* approach, 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 analyses of the generalization of QA models. 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) shows 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 because 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 $k$ QA agents. Each agent $i$ is trained on domain $dom_i$ and predicts an answer $Ans_i$. Without loss of generalizability, we assume that each agent is trained on a different domain and each question belongs to one of these domains. We define two complementary tasks: i) domain selection (Domain Selection Networks, DomSeN, 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, which can also give correct answers. 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.

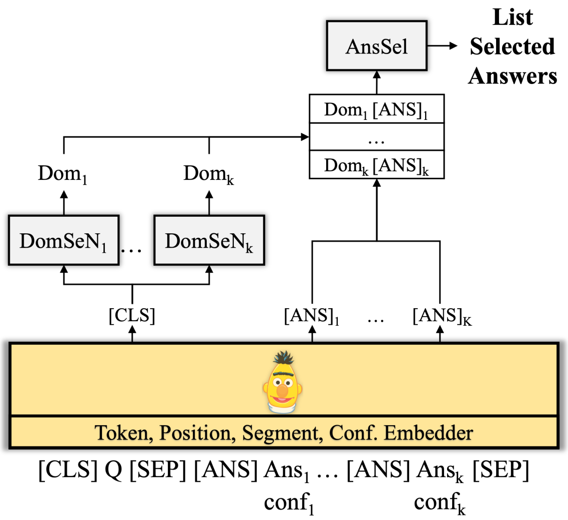


Figure 2: MetaQA architecture. The Domain Selection Networks, DomSeN, identifies the domain of input question Q. Answer Selection, AnsSel, selects the correct answers. $\text{conf}_k$ is the confidence score from the agent $k$ for answer $k$.

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.

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

The new $c_i$ is obtained with a feed-forward network $f$ that takes an answer confidence $\text{conf}_i$ and creates an embedding $c_i$.

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

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

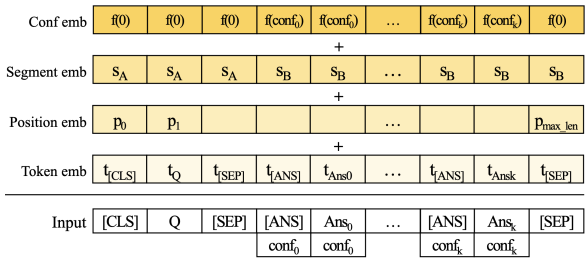


Figure 3: Description of our novel embedding system including confidence scores from the agents.

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. It is used as the input to $k$ independent feed-forward networks called *Domain Selection Network* (DomSeN) to identify the domain of the input question (i.e., the dataset from which the question comes) in a similar way as TWEAC. The second type of embedding used is the embedding of the `[ANS]` tokens, which contain the cues needed to identify the correct answers to the input question. These `[ANS]` embeddings are concatenated with the score of each corresponding domain $Dom_i$ and input into a final feed-forward network, called *Answer Selection* (AnsSel), that selects the correct answers according to the domain of the question and the candidate answers.

### 3.1 Training

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

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

$$\ell_{AnsSel} = \frac{1}{k} \sum_{i=0}^{k} CE(\hat{Ans}_i, y_i) \tag{4}$$

where $\ell_{DomSeN_i}$ is the loss of one DomSeN network and $\ell_{AnsSel}$ the loss of the AnsSel network. $\ell_{AnsSel}$ is the average of the cross-entropy loss $CE$ of each answer prediction $\hat{Ans}_i = \{0, 1\}$. Lastly, DomSeN networks use the Binary Cross Entropy.

We obtain the labels of AnsSel, $y_i$, by comparing the string prediction of each agent with the correct answer. If the F1 score is higher than a threshold, $\theta$, we consider the prediction as correct. As for DomSeN$_i$, 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. 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 include these two additional datasets to add more diversity. 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. Lastly, we include abstractive QA following (Khashabi et al., 2020) and a multimodal dataset 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). Then, we conduct hyperparameter tuning on the dev set and hypothesis testing on the test set. A summary of the datasets is available in Appendix A.1.

## 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.3. For the remaining datasets, we use agents that are publicly available on HuggingFace or Github with a performance close to the current state of the art. A summary of them is provided in Appendix A.2.

## 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 topics (or types of questions) to identify agents trained on that type of data (Geigle et al., 2021) and the simple max-voting ensemble. The second family of models is composed of 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 include the expert agents to analyze whether MetaQA closes the gap to them compared to the baselines.

## 4.4 Evaluation

Since MetaQA may select more than one answer, we select the answer with the highest confidence score by MetaQA as the decision of the model to evaluate it. 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 and Discussions

In this section, we answer the 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 Comparison with the Baselines

### 5.1.1 TWEAC

MetaQA outperforms TWEAC in all datasets except HellaSWAG and SIQA, as shown in Table 1.

| Dataset | MetaQA | TWEAC | Exp. Agent | UnifiedQA | MultiQA | Voting |
|---|---|---|---|---|---|---|
| SQuAD | 91.98±0.11† | 89.09±0.36 | 92.92 | 90.81 | **93.14±0.18** | 90.73 |
| NewsQA | 71.71±0.21† | 66.86±0.75 | **73.68** | 65.57 | 73.59±0.60 | 66.60 |
| HotpotQA | 79.27±0.15† | 74.96±0.59 | 80.60 | 77.92 | **81.68±0.22** | 71.71 |
| SearchQA | **81.98±0.25**†‡ | 80.41±0.22 | 81.04 | 81.61 | 80.45±1.82 | 68.87 |
| TriviaQA-web | **80.63±0.26**†‡ | 76.55±0.15 | 79.34 | 72.34 | 77.76±4.15 | 75.73 |
| NQ | 81.20±0.18† | 78.06±0.37 | 81.97 | 75.58 | **82.57±0.30** | 72.25 |
| DuoRC | **51.24±0.20**†‡ | 44.28±0.23 | 43.77 | 34.65 | 46.99±0.15 | 50.94 |
| QAMR | 83.78±0.14† | 78.77±0.48 | 84.00 | 82.70 | **84.62±0.14** | 73.07 |
| BoolQ | 73.14±0.23† | 72.20±0.03 | 72.17 | **81.34** | n.a. | 73.88 |
| CSQA | **78.66±0.19**† | 77.18±0.18 | 78.56 | 58.43 | n.a. | 68.41 |
| HellaSWAG | 73.19±1.01 | 77.12±0.30 | **77.14** | 36.01 | n.a. | 69.33 |
| RACE | 84.71±0.05† | 83.02±0.27 | **84.78** | 69.65 | n.a. | 67.30 |
| SIQA | 74.17±0.64 | 75.39±0.05 | **75.44** | 61.62 | n.a. | 70.01 |
| DROP | 73.04±1.98† | 69.12±0.36 | **74.61** | 42.45 | n.a. | 26.18 |
| NarrativeQA | **67.19±0.00** | **67.19±0.00** | **67.19** | 57.82 | n.a. | **67.19** |
| HybridQA | **50.94±0.00** | **50.94±0.00** | **50.94** | n.a | n.a | **50.94** |

Table 1: 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.

On average, MetaQA achieves an average performance boost of 2.42 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.

The reason for these results is that TWEAC only aims to identify the agent trained on the domain of the question while we retrieve the best answer prediction, even if it comes from out-of-domain models. For instance, in DuoRC, 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. In this way, MetaQA establishes a collaboration between agents.

We also observe that the gap between MetaQA and TWEAC is more significant on extractive QA than on multiple-choice. This is expected due to our selection of multiple-choice datasets. The substantial differences in the format of these datasets limit the potential agent collaboration. For instance, BoolQ is the only boolean dataset, and therefore, it can only be used to solve boolean questions, which do not appear in the other multiple-choice datasets. Also, SIQA, a commonsense reasoning dataset, uses a short context passage while CSQA (commonsense too) does not have any context, and hence, an agent trained for CSQA cannot be used successfully on SIQA. These characteristics of the setup makes the upper-bound performance of MetaQA to be the same as the expert agents. Yet, even with these limitations, MetaQA outperforms TWEAC in three of the five datasets. Also, the expert agents only significantly outperform MetaQA on 2/5 datasets. Lastly, the performance in NarrativeQA and HybridQA is the same because there is only one agent per dataset.

#### 5.1.2 UnifiedQA

MetaQA outperforms UnifiedQA by a striking 8.89. This is because of the limitations of UnifiedQA's 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

| Dataset | NewsQA | HotpotQA | SearchQA | TriviaQA | NQ | DuoRC | QAMR | CSQA | HellaSWAG | SIQA | DROP | Δ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 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 | 62.26 | 69.41 | 66.59 | 75.02 | 67.51 | **50.51** | 72.20 | 58.59 | 52.13 | 59.28 | 22.14 | - |
| OOD TWEAC | 57.65 | 43.98 | 57.93 | 66.62 | 65.37 | 47.32 | 69.59 | 47.46 | 50.59 | 59.16 | 20.53 | -6.31 |
| OOD UnifiedQA | 60.12 | 62.21 | 63.02 | 69.33 | 61.49 | 32.84 | 70.07 | 50.57 | 29.35 | 44.93 | 22.30 | -8.12 |
| OOD MultiQA* | **63.36** | **69.44** | **67.94** | **76.09** | **68.52** | 49.89 | **72.53** | n.a. | n.a. | n.a. | n.a. | 0.61 |
| OOD Max Voting | 63.25 | 67.59 | 61.76 | 73.81 | 68.27 | 50.48 | 68.92 | **58.94** | **64.03** | **63.22** | **22.46** | 0.64 |

Table 2: 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. Δ is the average performance gap to OOD MetaQA. * MultiQA uses a pseudo-OOD setup, see remarks in §5.2.

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.

#### 5.1.3 MultiQA

MultiQA slightly outperforms MetaQA by an average of 0.24. However, these gains are insignificant compared to its restrictions. MultiQA is only compatible with extractive QA (§4.3), while MetaQA is compatible with any QA format. Moreover, our model was trained on only 13% of its training set, as later discussed in §5.4. Furthermore, we observe that MultiQA mostly outperforms expert agents on Wikipedia-based datasets (i.e., SQuAD, HotpotQA, NQ, and QAMR). This might suggest that MultiQA is overfitted to Wikipedia due to its training on multiple datasets using Wikipedia paragraphs[2] and that would explain why it struggles with other minority domains. On the other hand, MetaQA excels in minority domains where it achieves a striking 4.15 points performance boost on DuoRC, 2.73 on TriviaQA-web, 1.55 on SearchQA, and in overall outperforms MultiQA by an average of 2.88. These results show the superior ability of MetaQA to avoid overfitting to a specific domain.

#### 5.1.4 Max-Voting

Lastly, MetaQA also outperforms max-voting by an average of 8.54. In the case of easy datasets such as SQuAD, the performance is similar because all expert agents excel in this dataset, so any approach to combine the agents would yield similar results. More interestingly, the performance in DROP is clearly far from MetaQA. We attribute this to the low performance of the extractive agents in this dataset and their similar wrong answers.

[2] MultiQA is trained on question and contexts (Wikipedia paragraphs). However, MetaQA does not have access to these paragraphs as shown in Figure 2.

### 5.2 Leave-One-Out Ablation

In this experiment, we analyze whether the combination of expert agents can successfully solve an out-of-domain (OOD) dataset. We conduct a leave-one-out ablation test in both MetaQA and the baselines. In the case of MetaQA, we remove the expert agent of the target dataset, retrain MetaQA again without this dataset, and evaluate it on the target dataset. Similarly, we retrain TWEAC, UnifiedQA, and MultiQA without the target dataset and evaluate the model on the target dataset. Lastly, we also use the Max-Voting baseline without the agent trained on the target dataset. We trained MetaQA five times with different random seeds for each target dataset and report their average results. However, we could not do this for the other models due to their much higher computation costs.

Table 2 shows that OOD MetaQA outperforms OOD TWEAC in all datasets by an average of 6.31. The larger gap in OOD than in in-domain scenarios (Table 1) supports our hypothesis: the topic-classification approach of TWEAC disregards high-performing models in OOD, and our solution of establishing a collaboration between the agents is able to combine skills.

OOD MetaQA also outperforms OOD UnifiedQA by a striking average of 8.13 points. In addition, in four datasets (TriviaQA-web, DuoRC, CommonsSenseQA, and HellaSWAG), the ablated MetaQA even outperforms the full UnifiedQA trained on those datasets. This further supports our approach of combining multiple agents, instead of datasets, in scenarios with a wide variety of domains and formats, where flexibility is key.

In the particular case of MultiQA, as discussed in §5.1.3, half of its training sets are based on Wikipedia paragraphs. Therefore, removing a Wikipedia-based dataset such as HotpotQA does not remove Wikipedia contents from its training

| 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 3: 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).

set[3]. As a consequence, this compromises the OOD setup. However, even under this pseudo-OOD setup, MultiQA only outperforms MetaQA by a slight margin of 0.61.

Lastly, we analyze the Max Voting baseline in this scenario. Although prior works disregard this baseline, the results in Table 2 show that OOD Max Voting outperforms all the other baselines and has a similar performance to OOD MetaQA. Its average gain with respect to OOD MetaQA is 0.64. However, this is not the overall trend. OOD MetaQA outperforms OOD Max Voting in 5/8 extractive QA datasets by a considerable margin of 3.19. On the other hand, multiple-choice datasets, especially the difference in HellaSWAG, incline the average towards OOD Max Voting. Despite the promising claims of prior works (Talmor and Berant, 2019; Khashabi et al., 2020) about OOD performance, these results suggest that aggregating a wide range of QA skills for different formats and domains in out-of-domain scenarios is still an open problem and non-neural baselines have strong results. Similar results have also been observed in retrieval methods, where non-neural baselines outperform supervised methods on OOD scenarios (Thakur et al., 2021).

### 5.3 Qualitative 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 types of DuoRC and NewsQA. DuoRC's questions are crowdsourced and are predominately *who-questions* (42% of the training set as shown in Appendix 9). 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 datasets 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 3.

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 3.

In SearchQA, the most helpful OOD agent is TriviaQA (5% chosen rate). This might be due to their similarities (Table 6). 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, and in those cases, the OOD agent was able to identify the exact answer. This is another example of the successful agent collaboration achieved by our MetaQA. Even though the in-domain agent almost has the correct answer, MetaQA selects an OOD agent that gives a better answer, as shown in the last row on Table 3.

### 5.4 Efficiency of MetaQA

We trained MetaQA with bins of QA instances for each dataset and observed 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

[3] This is not the case for MetaQA because our input is only the questions, answer predictions, and confidence scores, not the Wikipedia paragraphs.

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[4], 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 necessary 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 4 shows that removing the loss of the Domain Selection Network (DomSeN) 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.

| Model | Avg. Downgrade |
|---|---|
| $-\ell_{DomSeN}$ | -0.45 |
| – Conf. Emb. | -0.46 |

Table 4: Average performance loss across all datasets of each ablated model compared to the full model.

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

## 6 Conclusions

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 thanks to the agent collaboration established. Additionally, since MetaQA learns to match questions with answers instead of end-to-end QA, it is highly data-efficient to train. We leave as future work: i) combining partially correct answer predictions to generate a better one, ii) adding new agents without retraining MetaQA by fixing most of the weights and only training the weights of the new Domain 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 content given that the expert agents it aggregates are fair because MetaQA does not generate content. Rather it selects from Expert Agents. The datasets we use are well-known to be safe for research purposes and do not contain any personal information or offensive content. We also comply with the licenses and intended uses of each dataset. The licenses of each dataset are shown in Appendix A.1. We are not held responsible for errors, false or offensive content generated by the agents. MetaQA should be used at the users' discretion. Future work should address how to identify unfair or false content to avoid selecting it.

## Limitations

The main limitation of MetaQA is that when no agent has a correct answer, it returns an incorrect answer. Table 5 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 multiple-choice datasets (without BoolQ, another outlier). As for NarrativeQA and HybridQA, there are many unsolvable questions because we only use one agent for each of them and these agents have a relatively low performance.

Also, if the agents do not fit in memory at the same time, it would be necessary to run them sequentially, which would increase the inference time.

| 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 5: Percentage of unsolvable questions for our MetaQA with the selected agents, i.e., none of the agents can give a correct answer.

Yet, it might be possible to overcome this limitation 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). This would allow us to skip some agents at run-time and improve the running time dramatically in low-memory scenarios.

## 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.

Subhabrata Mukherjee, Ahmed Hassan Awadallah, and Jianfeng Gao. 2021. Xtremedistiltransformers: Task transfer for task-agnostic distillation.

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.

Nandan Thakur, Nils Reimers, Andreas Rücklé, Abhishek Srivastava, and Iryna Gurevych. 2021. BEIR: A heterogeneous benchmark for zero-shot evaluation of information retrieval models. In *Thirty-fifth Conference on Neural Information Processing Systems Datasets and Benchmarks Track (Round 2)*.

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 Datasets

Table 6 summarizes the characteristics of the datasets, contains the size of the train, validation, and test splits of each dataset, and their licenses. In the case of RACE, the authors did not provide any license but specified that it could only be used for non-commercial research purposes. In the case of CommonSenseQA and SIQA there is no license specified, but they are freely available to download. Therefore, our use of these datasets complies with their licenses and intended uses.

## A.2 Expert Agents

Table 7 provides the links to download the expert agents used in this work. In the case of NarrativeQA and HybridQA, we only employ one agent because of the difficulty of obtaining others. Both of these datasets use uncommon modalities (abstractive and table+text). Therefore, it is not straightforward to adapt other models to these datasets.

## A.3 Implementation

Our model was implemented using PyTorch (Paszke et al., 2019) and HuggingFace's Transformers library (Wolf et al., 2020). 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 an output size of 1024 (i.e., the hidden size of Span-BERT Large). $\alpha_1$ and $\alpha_2$ in Eq. 3 are set to 0.5 and 1 respectively. The Domain 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 the output of the domain selection network). The threshold $\theta$ 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.

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

Lastly, the max-voting baseline assumes that two answers are the same if the F1 score is higher than a threshold (0.9). We tuned this parameter on the dev set searching in the range $[0.5, 0.6, ..., 1.0]$. We used the implementation of HuggingFace's SQuAD F1 metric[5]. In the case that two answers have the same amount of votes, we select the one with the highest confidence score given by an agent.

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[6].

All the experiments were conducted in a SLURM cluster where each job was assigned to different computer nodes with different CPUs and GPUs. Therefore, comparing the running time of each model is not possible.

## A.4 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.5 MetaQA on a Single Dataset

We conduct an additional experiment to analyze the behavior of MetaQA with multiple expert agents trained in a single dataset. We train MetaQA for three NewsQA agents: RoBERTa-base, XtremeDistil (Mukherjee et al., 2021), and SpanBERT, and evaluate it on NewsQA. As observed in Table 8, MetaQA performs on par with the agents. However, the performance gap is smaller than in the main use case (§5.1). This is attributed to the similarities between the models. These three models are all Transformers and

[5]https://huggingface.co/metrics/squad
[6]https://pypi.org/project/rouge-score/

| | Dataset | Characteristics | Train | Dev | Test | License |
|---|---|---|---|---|---|---|
| Extractive | SQuAD (Rajpurkar et al., 2016) | Crowdsourced questions on Wikipedia | 6573 | 5253 | 5254 | MIT |
| | NewsQA (Trischler et al., 2017) | Crowdsourced questions about News | 74160 | 2106 | 2106 | MIT |
| | HotpotQA (Yang et al., 2018) | Crowdsourced multi-hop questions on Wikipedia | 72928 | 2950 | 2951 | MIT |
| | SearchQA (Dunn et al., 2017) | Web Snippets, Trivia questions from J! Archive | 117384 | 8490 | 8490 | MIT |
| | NQ (Kwiatkowski et al., 2019) | Wikipedia, real user queries on Google Search | 104071 | 6418 | 6418 | MIT |
| | TriviaQA-web (Joshi et al., 2017) | Web Snippets, crowdsorced trivia questions | 61688 | 3892 | 3893 | MIT |
| | QAMR (Michael et al., 2018) | Wikipedia, predicate-argument understanding | 50615 | 18908 | 18770 | MIT |
| | DuoRC (Saha et al., 2018) | Movie Plots from IMDb and Wikipedia | 58752 | 13111 | 13449 | MIT |
| Multiple-Choice | RACE (Lai et al., 2017) | Exams requiring passage summarization and attitude analysis | 87866 | 4887 | 4934 | NA |
| | CSQA (Talmor et al., 2019) | Web Snippets, common-sense reasoning | 9741 | 611 | 610 | NA |
| | BoolQ (Clark et al., 2019) | Wikipedia, Yes/No questions | 9427 | 1635 | 1635 | CC BY-SA 3.0 |
| | HellaSWAG (Zellers et al., 2019) | Completing sentences using common sense | 39905 | 5021 | 5021 | MIT |
| | SIQA (Sap et al., 2019) | Common sense in social interactions | 33410 | 977 | 977 | NA |
| Abs. | DROP (Dua et al., 2019) | Wikipedia, numerical reasoning | 77409 | 4767 | 4768 | CC BY-SA 4.0 |
| | NarrativeQA (Kočiský et al., 2018) | Books, Movie Scripts | 32747 | 3461 | 10557 | Apache 2.0 |
| MM | HybridQA (Chen et al., 2020) | Wikipedia tables and paragraphs | 62682 | 1733 | 1733 | MIT |

Table 6: Summary of the datasets used. Abs. stands for abstractive and MM for multi-modal.

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

Table 7: List of the expert agents, datasets in which they are used, and links to download.

trained on the same dataset, so it is natural that they are similar. An approach such as MetaQA excels when the agents are very different, as in Table 1, where the agents were trained on different datasets and therefore have different skills.

| Model | F1 Score |
|---|---|
| MetaQA | **73.73** |
| SpanBERT | 73.68 |
| RoBERTa | 73.15 |
| XtremeDistil | 64.16 |

Table 8: MetaQA trained only on NewsQA agents.

### A.6 Wh-word Statistics

Table 9 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 9: Statistics of wh-words per dataset.