ELI5: Long Form Question Answering

Angela Fan[1,2] **Yacine Jernite**[*1] **Ethan Perez**[*3]
David Grangier[4] **Jason Weston**[1] **Michael Auli**[1]
[1]Facebook AI Research [2]LORIA [3]NYU ‡ [4]Google AI ‡
[angelafan,yjernite,jase,michaelauli]@fb.com,
perez@nyu.edu, grangier@google.com

Abstract

We introduce the first large-scale corpus for long-form question answering, a task requiring elaborate and in-depth answers to open-ended questions. The dataset comprises 270K threads from the Reddit forum "Explain Like I'm Five" (ELI5) where an online community provides answers to questions which are comprehensible by five year olds. Compared to existing datasets, ELI5 comprises diverse questions requiring multi-sentence answers. We provide a large set of web documents to help answer the question. Automatic and human evaluations show that an abstractive model trained with a multi-task objective outperforms conventional Seq2Seq, language modeling, as well as a strong extractive baseline. However, our best model is still far from human performance since raters prefer gold responses in over 86% of cases, leaving ample opportunity for future improvement.[1]

Question: How do Jellyfish function without brains or nervous systems? [...] (60 words)

Answer: Jellyfish may not have a brain, but they have a rough nervous system and innate behaviours. However, they are very simple creatures. They're invertebrate: creatures without a backbone. Most jellyfish have really short life spans. Sometimes just a couple of hours. [...] As their name implies, they are largely composed of basically jelly inside a thin membrane. They're over 95% water. (327 words)

Documents: [...] Jellyfish do not have brains, and most barely have nervous systems. They have primitive nerve cells that help them orient themselves in the water and sense light and touch. [...] While they dont possess brains, the animals still have neurons that send all sorts of signals throughout their body. [...] They may accomplish this through the assistance of their nerve rings. Jellyfish don't have brains, and that's just where things begin. They don't have many of the body parts that are typical in other animals. [...] (1070 words)

Figure 1: ELI5 example. Models must write multi-sentence answers given questions and supporting web documents.

1 Introduction

Existing question answering datasets have enabled significant progress in models that provide extractive or unambigious short answers. However, less attention has been paid to open-ended questions that require explanations. In this work, we present ELI5: a Long Form Question Answering dataset that emphasizes the dual challenges of isolating relevant information within long source documents and generating paragraph-length explanations in response to complex, diverse questions (see illustrations in Figures 1 and 2).

The first challenge of ELI5 is the length and diversity of answers that span multiple sentences: questions are complex and cannot be easily addressed by a short response (Nguyen et al., 2016) or by extracting a word or phrase from an evidence document (Rajpurkar et al., 2016). Answers also represent one of several valid ways of addressing the query. Many state-of-the-art question answering models perform well compared to human performance for extractive answer selection (Radford et al., 2018; Devlin et al., 2018). However, their success does not directly carry over to our setting.

The second challenge is the length and diversity of the content from knowledge sources required to answer our questions. We leverage evidence queried from the web for each question. In contrast to previous datasets where the human written answer could be found with lexical overlap methods (Weissenborn et al., 2017), ELI5 poses a significant challenge in siphoning out important information, as no single sentence or phrase contains the full answer. While there are some datasets that do require multi-sentence supporting knowl-

[*] Equal contribution
‡ Work done while at Facebook AI Research
[1]Dataset, Pretrained Models, and Additional Information is available: https://facebookresearch.github.io/ELI5, https://github.com/facebookresearch/ELI5

Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, pages 3558–3567
Florence, Italy, July 28 - August 2, 2019.

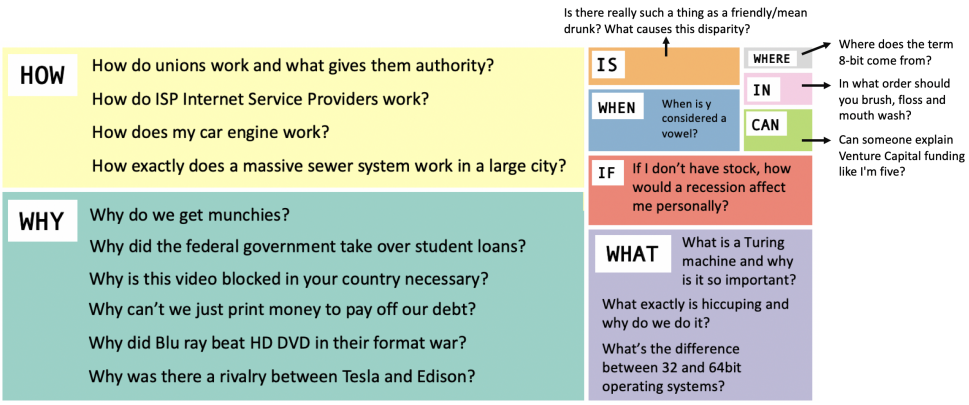

Figure 2: ELI5 questions by starting word, where box size represents frequency. Questions are open ended and diverse.

edge such as TriviaQA (Joshi et al., 2017), their answers are still short.

We benchmark the performance of several extractive, retrieval, and generative models. Evaluation of our task, and of multi-sentence text generation in general, is challenging. We draw upon several evaluation metrics that quantify performance on intermediary fill-in tasks that lead up to the full answer generation. The overall answer generation quality is measured with ROUGE (Lin, 2004) and various human evaluation studies.

We develop a strong abstractive baseline by training a Seq2Seq model on multiple tasks over the same data: language modeling, masked word prediction (Devlin et al., 2018) and answer generation. We show this approach outperforms conventional Seq2Seq and language modeling, as well as a strong extractive baseline based on BidAF (Seo et al., 2017) but generalized to multi-sentence output. However, our best-performing model is still far from the quality of human written answers, with raters preferring the gold answers 86% of the time. Further, we show that model performance is strongly limited by the ability to comprehend long multi-document input and generate long outputs to form a comprehensive answer, leaving this challenge for future research.

2 Related Work

Various QA datasets have been proposed in roughly two categories: extractive answers and short abstractive answers (see Table 1).

Extractive QA Extractive question answering datasets such as TREC (Voorhees, 2003), SQuAD (Rajpurkar et al., 2016, 2018), NewsQA (Trischler et al., 2017), SearchQA (Dunn et al., 2017), and QuAC (Choi et al., 2018) constrain the answer to a word or short phrase from the input and evaluate using exact match or F1 with the ground truth span. HotpotQA (Yang et al., 2018) extends this approach by building questions which challenge models to conduct multi-hop reasoning across multiple paragraphs, but the answer is still a short span. Further, the answer must be straightforward, as it needs to be copied from the supporting evidence — precluding most "how" or "why" type questions.

Abstractive QA Abstractive datasets include NarrativeQA (Kocisky et al., 2018), a dataset of movie and book summaries and CoQA (Reddy et al., 2018), a multi-domain dialogue dataset. Both collect responses with crowdworkers and find that written answers are mostly extractive and short. MS MARCO (Nguyen et al., 2016), a dataset of crowdsourced responses to Bing queries, has written answers around 1 sentence long with short input passages. TriviaQA (Joshi et al., 2017) contains longer multi-document web input, collected using Bing and Wikipedia. As the dataset is built from trivia, most questions can be answered with a short extractive span.

Multi-document summarization The ELI5 task of writing a paragraph length response from multiple supporting documents can be seen as a form of query-based multi-document summarization (Tombros and Sanderson, 1998). Summarization tasks such as DUC 2004[2] involve long input and multi-sentence generation, but contain much less training data compared to ELI5. WikiSum (Liu et al., 2018) proposes writing Wikipedia articles as a multi-document summarization task. ELI5 requires more directed

[2] https://duc.nist.gov/duc2004/

Dataset	Average # of Words			1st Question Word Frequency (%)								
	Question	Document(s)	Answer	Why	How	What	When	Where	Who	Which	OTHER	# Q-A Pairs
ELI5	**42.2**	857.6 (212K)	**130.6**	**44.8**	**27.1**	18.3	**11.3**	2.0	1.8	0.8	6.1	**272K**
MS MARCO v2 (Nguyen et al., 2016)	6.4	56	13.8	1.7	16.8	35.0	2.7	3.5	3.3	1.8	35.3	183K
TriviaQA (Joshi et al., 2017)	14	**2895**	2.0	0.2	3.9	32.6	2.0	2.1	16.8	**41.8**	0.6	110K
NarrativeQA (Kocisky et al., 2018)	9.8	656	4.7	9.8	10.7	38.0	1.7	**7.5**	**23.4**	2.2	6.8	47K
CoQA (Reddy et al., 2018)	5.5	271	2.7	2	5	27	2	5	15	1	**43**	127K
SQuAD (2.0) (Rajpurkar et al., 2018)	9.9	116.6	3.2	1.4	8.9	**45.3**	6.0	3.6	9.6	4.4	17.6	150K
HotpotQA (Yang et al., 2018)	17.8	917	2.2	0.1	2.6	37.2	2.8	2.2	13.8	28.5	12.8	113K

Table 1: Comparing large-scale QA datasets. ELI5 has answers an order of magnitude longer and more open-ended questions.

text generation to answer a question, rather than to write about a general topic. In addition, ELI5 contains a diverse set of questions which can involve more than one Wikipedia concept.

3 Making a Long Form QA Dataset

3.1 Creating the Dataset from ELI5

There are several websites which provide forums to ask open-ended questions such as Yahoo Answers, Quora, as well as numerous Reddit forums, or subreddits. We focus on the subreddit *Explain Like I'm Five* (ELI5) where users are encouraged to provide answers which are comprehensible by a five year old.[3] ELI5 is appealing because answers are supposed to be entirely self contained, and thus rely less on pre-existing knowledge of the world and use simpler language that is easier to model.

Questions and answers. We select a set of questions and answers from the ELI5 forum up to July 2018 and then filter it based on how users rated these pairs. First, we only retain questions which have a score of at least two, that is two more 'upvotes' than 'down-votes'. Second, there must be at least one answer with a score of at least two. This yields a final number of 272K questions, and ensures that at least one person other than the author has read the thread and deemed it appropriate. For each thread, we select the answer with the highest voting score as the reference. Note that 63% have one or more other valid answers by our upvote criteria, potentially doubling the size of the available training data.

Preparing supporting information. Next, we collect web sources for every question to provide relevant information that a system can draw upon when generating an answer. Wikipedia has been found effective for factoid-oriented questions (Joshi et al., 2017; Chen et al., 2017). However, early experiments in our setting showed it to be insufficient to cover the wide range of topics present in ELI5 and to address the open-ended nature of the questions. Instead, we use web data provided by Common Crawl.[4] Specifically, we consider each of the individual pages in the July 2018 archive (roughly one per URL) as a single document. The data is tokenized with Spacy[5] and we select English documents with FastText language identification (Bojanowski et al., 2017). Finally, we index the data with Apache Lucene.[6]

Creating support documents. We query the index for the 272K questions and gather the 100 most relevant *web sources* for each question, excluding Reddit. Each *web source* is the extracted text of one page in Common Crawl. This leads to supporting text for each question of a few hundred thousand words. There is a good chance that the supporting text contains the necessary information to answer the question, but the sheer amount of data is far beyond the scope of what many modern models can handle. We therefore filter the 100 web sources by selecting specific passages using a simple heuristic: we split each web source into sentences, find sentences with the highest TFIDF similarity with respect to the question, add some local context for each of these, and concatenate the result into a single *support document*, with special tokens indicating non-contiguous passages and document shifts. Each *support document* is the result of this processing to concatenate relevant information from the web sources.

We find that extracting 15 passages with a context of one sentence before and after the initial selection provides the best trade-off between support document length and likelihood of containing relevant information, where relevance is measured as the likelihood of containing a sentence which has

[3] https://www.reddit.com/r/explainlikeimfive

[4] http://commoncrawl.org

[5] https://spacy.io

[6] http://lucene.apache.org

% Correct Human Answers	94.5
% Correct Human Answers with Explanation	90.2
% Support Document contains Answer	65.0
% Support Document contains Relevant Info	92.0

Table 2: Annotated subset of ELI5 to assess answerability.

high ROUGE with the answer. We release all 100 Common Crawl IDs for each question and a script to create the support document so future research can use the support document or choose to further investigate the information retrieval problem.

Finalizing the data set. If the training data contains questions that are too similar to the validation and test data, a model may perform well on these examples by memorizing related examples. We prevent this by building the validation and test set to contain questions that are sufficiently different from the training data. We compute the TFIDF similarity between each pair of questions in the entire dataset and sample the validation and test set from the subset which has no close neighbor by TFIDF score. The final dataset contains 237K train examples, 10K for valid, and 25K for test.

3.2 Dataset Analysis

Table 1 compares ELI5 to related datasets in terms of the length of the question, support document, answer, as well as statistics on the question types.

First, ELI5 questions are much longer than in other datasets. This is because the initial question is often followed by a clarifying paragraph detailing what aspect of the general theme should be addressed or the question's starting assumptions, which need to be considered to answer well. To get a rough idea of the different questions, we categorize them based on interrogative words. ELI5 focuses on open-ended queries which are less represented in other extractive or abstractive datasets. Figure 2 shows examples of ELI5 questions split by type and Appendix Figure 11 displays random examples from the ELI5 training set. Interestingly, even *What* questions tend to require paragraph-length explanations (*What is the difference...*).

Support documents contain 22-60 sentences or on average 858 words, which puts ELI5 on the higher end of published datasets for document length. ELI5 contains long-form answers with an average length of 6.6 sentences, or 130 words.

Next, we analyze a random subset of ELI5 to assess the feasability of answering the questions in the dataset. We judge if the question is answerable by reading each question, the gold answer, and the support document we have created with TF-IDF extraction. Note that questions can have multiple parts and all parts of the question must be answered. We sample 500 randomly question-answer pairs from the training set and find that 94.5% of gold answers fully address the question (Table 2) based on the information in the support document. Figure 12 in Appendix F displays examples of human answers that do not correctly answer the question. A small proportion of answers are correct but do not explain the answer. On the support document side, 65% of the support documents we construct provide the answer to the question, and 92% of support documents provide information relevant to the question.

4 Evaluation Methods

Evaluating long-form answers. There are several aspects to quality: answers should be topical and accurate, fluent, and coherent from start to end. We judge the accuracy aspect by comparing to the gold answer. ROUGE (Lin, 2004) measures similarity between a model output and one or several references, and is often used in summarization. While our task presents different challenges, such as the diversity of possible answers to a question, we still find the corpus-level metric to be useful to rank different related models (§6). We report F1 for ROUGE-1, ROUGE-2, and ROUGE-L.

Abstractive model metrics. For generative models, perplexity (PPL) measures the ability to predict the next word in a sequence given its context. For a variant which focuses on semantically important words, we report FILL-1, the accuracy at which models generate different Nouns, Verbs, and Adjectives given the correct preceding tokens in the first 2K examples of the test set. Finally, ROUGE-20% measures the model's ability to complete an answer given the first 80% of the reference answer, the question, and the support document. Specifically, we generate a number of tokens corresponding to 20% of the average answer length in the validation set, and measure ROUGE between these and the last 20% of the reference. We mentioned that there are several valid ways to answer most questions. This measure abstracts away this variability and evaluates a system's ability to complete an answer.

Extractive: But none of these hypotheses make much sense for a jellyfish because they do nt have brains at all. They just have a nerve net — a loose ring of neurons that runs around the rim of their pulsating bells. They have an unusual nervous system, because jellyfish are not bilaterally symmetrical — that is, they dont have a left side and a right side. Jellyfish don't have brains, but their nervous systems detect smells, light and other stimuli, and they coordinate their physical responses.

Language Model: They don't have brains. They have a nervous system. They do not have a brain. They are not conscious. They can not move. They move. Their nervous system is not a muscle. They use their muscles to move. The nervous system does not have any nerves. They just have a nerve. They also have a muscle that is very similar to a muscle in that it is very much like a muscle, but instead of a muscle it is a muscle and it is like a nerve, but it is not muscle. The nerve is like the muscle, it is just a muscle but it has a nerve and it has nerves. The nerves are like the muscles, but they have nerves.

Multi-task Seq2Seq: Jellyfish don't have brains. Jellyfish have a nervous system. They know how to swim, they don't even know what to do with their bodies. They have a "nervous system", which is basically a bunch of nerve cells that connect to the body. The nervous system is made up of neurons, neurons, nerves, etc. Each neuron doesn't have a specific set of "neurons", it just has a different set of nerves. It has a set of sensory neurons, which are all connected to each other. It just knows that it does react to the stimulus, and it knows that the stimulus is there. They have a bunch of nervous systems that act like a filter to get information back.

Figure 3: Example answers from the extractive BidAF model, Question + Document + Answer language model, and Multi-task Seq2Seq model for the question "How do Jellyfish function without brains or nervous systems?" (cf. Figure 1).

Human evaluation. We use crowdworkers to conduct three assessments. First, evaluators rate the *fluency* of human and model generated answers on a 5-point Likert Scale, from "very poorly written" to "easily readable" (500 evaluations). Second, evaluators are given question-answer pairs and are asked if the answer is *correct* (500 evaluations) [7]. We also evaluated a smaller subset ourselves while additionally looking at the support documents (100 evaluations) to assess answer accuracy. Lastly, crowdworkers are given the question and answers from two models and asked to decide which answer they *prefer* while considering readability and accuracy (1000 evaluations). Each crowdworker assessment is made by 3 different evaluators. The same questions are used for all models and must be at least 5 words long.

5 Models

5.1 Extractive and Retrieval Models

Retrieval baseline and oracle. We report ROUGE for a retrieval system that returns the answer of the closest question in the training set. Specifically, we perform a nearest neighbor search (Johnson et al., 2017) over the average word embeddings of the question using FASTTEXT (Bojanowski et al., 2017). We also compute an approximate oracle score for extractive systems by using the reference answer to select similar sentences from the support document to maximize ROUGE. Computing ROUGE between the reference and all sets of sentences from the source is intractable. Instead, we perform a beam search that adds sentences maximizing TFIDF with respect to the answer. The final beam is re-ranked using ROUGE with respect to the reference answer. We run this algorithm on our support document and on the full set of web sources for each validation and test question, selecting up to 10 sentences with a beam of size 10.

Extractive models. The first baseline we explore simply returns the 7 sentences from the support document which have the highest TFIDF similarity with the question. We also evaluate models which score sentences from the support document based on the question and return the highest scoring sentences in their original order (the number is tuned on the validation set to maximize ROUGE). We train a model based on BidAF (Seo et al., 2017). We create an extractive training set by finding the span of up to 5 contiguous sentences in the support document which have the highest ROUGE with respect to the reference answer, and sub-sample other support document sentences so that the final training document is shorter than 400 words. We then train a BidAF model to predict the extracted span in the sub-sampled support document based on the question. For test, we compute the span score for each individual sentence, and return the 5 with the highest score as it performed best compared to returning 3 or 7 sentences.

5.2 Abstractive Models

Language and Seq2Seq models. We train several models based on the Transformer architecture (Vaswani et al., 2017), both in its language model and sequence-to-sequence (Seq2Seq) con-

[7] We experimented with a variant where crowdworkers were allowed to select a third *I don't know* option, but found it was used only around 8% of the time.

Model	PPL	ROUGE		
		1	2	L
Support Document	-	16.8	2.3	10.2
Nearest Neighbor	-	16.7	2.3	12.5
Extractive (TFIDF)	-	20.6	2.9	17.0
Extractive (BidAF)	-	23.5	3.1	17.5
Oracle support doc	-	27.4	2.8	19.9
Oracle web sources	-	54.8	8.6	40.3
LM Q + A	42.2	27.8	4.7	23.1
LM Q + D + A	33.9	26.4	4.0	20.5
Seq2Seq Q to A	52.9	28.3	5.1	22.7
Seq2Seq Q + D to A	55.1	28.3	5.1	22.8
Seq2Seq Multi-task	32.7	28.9	5.4	23.1

Table 3: Comparison of oracles, baselines, retrieval, extractive, and abstractive models on the full proposed answers.

Model	FILL-1 acc.			ROUGE-20%		
	N	V	A	1	2	L
LM Q + A	31.0	29.6	20.6	26.5	7.0	21.1
LM Q + D + A	30.9	28.9	19.9	26.3	7.8	21.3
S2S Q to A	21.7	23.0	15.5	33.6	11.5	29.5
S2S Q + D to A	27.6	26.3	19.4	32.7	10.7	28.6
S2S Multi-task	27.9	26.7	19.9	37.2	14.6	33.0

Table 4: Intermediary fill-in tasks for sequential generation.

figurations. To investigate how much information from the document the model uses, we train a language model on the concatenation of *Question*, *Support Document*, and *Answer* (Q + D + A) as well as on the *Question* and *Answer* (Q + A). Similarly, one Seq2Seq configuration goes from Q to A, and the other from Q + D to A. In all cases, Q, D, and A are separated by special tokens.

Multi-task training. Language models are trained to predict all tokens in the question, web source, and answer. However, the standard Seq2Seq model only receives training signal from predicting the answer which is much less than the language model gets. This can contribute to learning poor quality representations compared to language models. To address this, we train a *multi-task* Seq2Seq model: during training, we multi-task between several generation tasks, including language modeling of Q + D + A by the decoder and variations of source/target pairs (see Appendix A). We add a masked word prediction task (Devlin et al., 2018) where 15% of tokens in the input are masked and must be recovered by the model in the correct order, and append a marker at the start of each sequence to indicate the task.

Data processing. To reduce the vocabulary, we apply byte-pair encoding (Sennrich et al., 2016) to generate 40K codes which are applied to all datasets. We model a vocabulary of 52,863 tokens for answer generation. We use the Transformer implementation of fairseq-py (Gehring et al., 2017) and train with the big architecture following the details in (Vaswani et al., 2017). Given our data length, we train with a large batch size by delaying gradient updates until a sufficient number of examples have been seen (Ott et al., 2018).

Generation. We generate from abstractive models using beam search with beam 5. We disallow repeated trigrams to prevent repetition, a technique commonly used in multi-sentence summarization (Paulus et al., 2017; Fan et al., 2018). For the full answer generation task, we tune a minimum and maximum length for generation on the valid set and apply these settings to the test set.

6 Results

6.1 Overview of Model Performance

Full answer ROUGE. Table 3 shows that the nearest neighbor baseline performs similarly to simply returning the support document which indicates that memorizing answers from the training set is insufficient. For extractive models, the oracle provides an approximate upper bound of 27.4 ROUGE-1. The BidAF model is the strongest (23.5), better than TFIDF between the question and the support document to select sentences. However, these approaches are limited by the support document, as an oracle computed on the full web sources achieves 54.8.

Abstractive methods achieve higher ROUGE, likely because they can adapt to the domain shift between the web sources and the ELI5 subreddit. In general, Seq2Seq models perform better than language models and the various Seq2Seq settings do not show large ROUGE differences. Figure 3 shows an example of generation for the language model and the best Seq2Seq and extractive settings (see Appendix F for additional random examples).

Perplexity and fill-in tasks. Tables 3 and 4 present metrics specific to sequential generation models: perplexity of the answer, accuracy of the model's FILL-1 word prediction for Nouns, Verbs, and Adjectives, and ROUGE of the conditional generation of the last 20% answer words. The language model perplexity is much lower than that of the standard Seq2Seq setting – this is likely linked to the number of output tokens the system

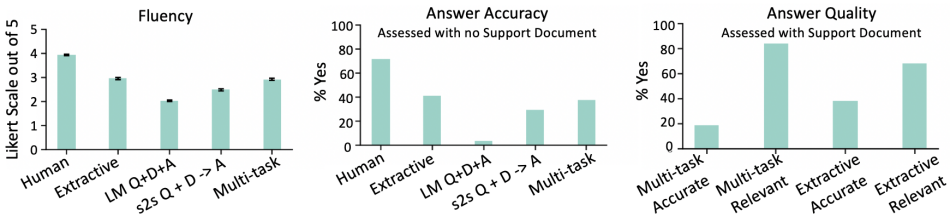

Figure 4: Human evaluation of answer fluency and accuracy — with and without access to supporting evidence documents

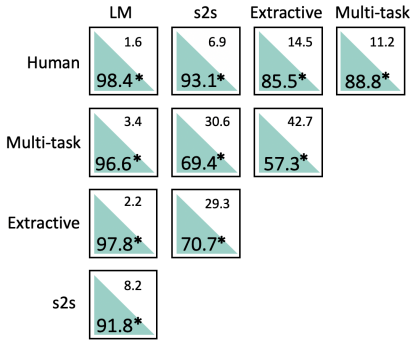

Figure 5: Human preferences for pairwise comparisons. The better model's % preference is bolded. * indicates statistical significance.

is required to predict at training time. The multi-task Seq2Seq experiment, in which the Seq2Seq decoder is trained to predict the question and the document, in addition to the answer, can reach the same perplexity as the language model. ROUGE-20% shows a much starker contrast between language modeling and Seq2Seq, as well as between standard Seq2Seq and multi-task training. The latter achieves strong performance of 37.2 ROUGE-1. However, both versions of the language model are still better at FILL-1. These results suggest that the Seq2Seq model is better than the language model in maintaining coherence and that Seq2Seq relies on information over many time steps.

Human evaluation. Human answers are rated highest in terms of fluency (Figure 4, left). The extractive model outputs human-written text which is likely fluent but with the failure mode of concatenating unrelated sentences. The multi-task model performs similarly to the extractive model which indicates that abstractive methods can generate coherent answers. The language model and standard Seq2Seq trail behind.

To get a sense of the stability of our results, we analyzed the standard deviation of three independent fluency trials conducted on separate days and we find low variation (Appendix E, Figure 10). We also measure agreement between crowdworkers in selecting positive (scores 4 and 5), negative (1 and 2), or neutral (3) choices on the 5-point Likert scale, and find that 2 crowdworkers agree almost 100% of the time (Appendix E, Figure 10).

In answer accuracy (Figure 4, middle), there is a large gap between human performance and all models. The language model is almost never accurate, while the extractive model is slightly more so than the multi-task model. Crowdworkers assessing accuracy do not have the support document. We evaluate accuracy ourselves with the support document in Figure 4, right. Similar to crowdworkers, we find 40% of extractive answers to be accurate. We find only 19% of multi-task model answers are fully accurate; even if the model output answers the question, it can generate a sentence with an incorrect statement. In contrast, the extractive model copies sentences from human-written text. However, the multi-task model is better at generating relevant answers (84% relevancy compared to 68% for extractive), as the extractive model is constrained by the support document.

Figure 5 presents pairwise preference judgments of human annotators shown answers from two of the five systems. The reference answer is preferred over the output of all of our trained models in at least 85.5% of cases, indicating there is substantial room for improvement. The multi-task abstractive setting comes next, closely followed by the extractive (multi-task is only preferred in 57% of comparisons), then the standard Seq2Seq and finally the language model, considered worse than any other setting in at least 91% of cases.

We use a two-tailed binomial test to test statistical significance of the pairwise judgments and it shows that all judgments are statistically significant at $p = 0.05$.

6.2 Quantitative and Qualitative Analysis

Discussion of the proposed metrics. We present a number of metrics which provide insight into various model behaviors. We recommend

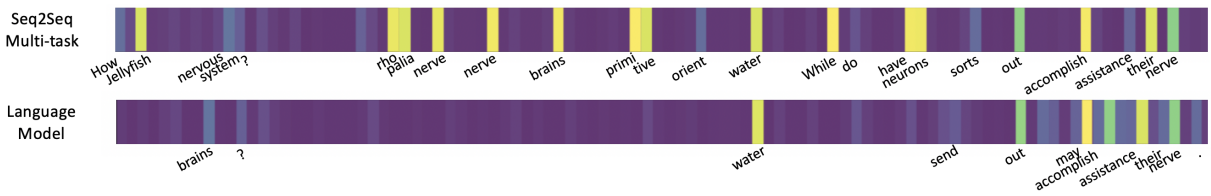

Figure 6: Attention over the question and supporting evidence for the Multi-task Seq2Seq model and Question + Document + Answer language model. Attention is shown for the first word of answer generation.

future work to report full ROUGE and ROUGE-20%. Perplexity and FILL-1 focus on local prediction and are poor indicators of overall appropriateness for the full task. Full answer ROUGE discriminates reasonably well between models with the same general architecture, but cannot rate an abstractive system against an extractive one. The ROUGE-20% measure abstracts away some variability and focuses on coherence between the beginning and end of an answer. This metric correlates with human judgments of quality but can only be reported for sequential generation.

Analysis of extractive, LM and Seq2Seq models. Language models perform better than Seq2Seq in terms of perplexity and FILL-1, while being significantly worse at ROUGE-20% and human evaluations. To investigate this, we visualize the attention mechanism at the start of answer generation in Figure 6. The attention of the language model is strongly focused on nearby context when generating the first word of the answer, whereas the multi-task Seq2Seq model attends more evenly to relevant information in the question and the document. This validates our assumption that the language model's focus on local context is insufficient for high quality answers.

In Figure 7 (left), we further investigate how the relevance and quality of the support document extraction step affects the answers provided by the extractive and abstractive setting. The ROUGE score is displayed for data subsets, partitioned by percentile of word overlap of the answer with the support document (e.g. how many answer words appear). While both models perform better for documents with higher ROUGE overlap between support document and human answer, the abstractive setting is much better at compensating for when the support document has lower relevance.

Data size and initial selection. There is a large difference between the extractive oracle ROUGE using our support document and the oracle on full

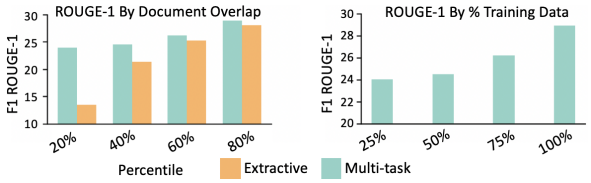

Figure 7: (left) Model score by document-answer similarity. (right) Seq2Seq multi-task score by amount of training data.

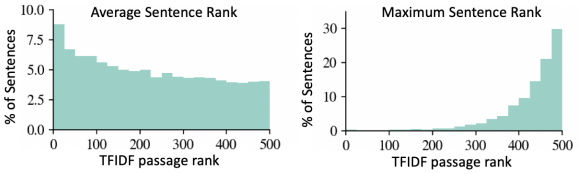

Figure 8: (left) TFIDF rank of source passage for oracle sentences. (right) Highest rank used per question.

web sources. This suggests that the initial selection of our support document severely limits access to relevant information. To assess the impact of support document size, we re-run the selection step for 1000 examples to extract 500 passages instead of 20, and run the oracle on these new inputs. Figure 8 shows the TFIDF rank of the passages from which sentences are selected. While slightly more sentences are extracted from the higher ranking passages, less than 9% come from the first 20, and most oracles have at least one sentence from the last 100. For a model to perform best, it would have to handle inputs tens of thousands of words long. In Table 3, we show an oracle computed on the full web sources has much higher ROUGE than an oracle computed on the support document.

We analyze the impact of data size on performance in Figure 7. We train the multi-task model on 25%, 50%, and 75%, and the all of the data to compare performance. ROUGE increases as a function of the data used and even though ELI5 is one of the larger QA datasets (§3), this shows that collecting more still helps. While we only used one reference answer per question here, recall that over half of them have multiple answers, which could be leveraged to train better models.

Combining challenges. Our task blends the inter-dependent challenges of retrieving information, reasoning, and writing long outputs. Studying each of these aspects in context is particularly important. For example, we show that the abstractive model's ability to compensate for a (realistically) imperfect support document is essential to its relative success over extractive methods. The fluency gap between the reference and the extractive system in human evaluation also suggests that the latter may require sequential decision capabilities. This kind of decision making is necessary to address the dual challenges of reasoning over several supporting facts and generating long coherent outputs. We see our task's need to combine complementary systems as critical to gaining insights into their individual behaviors.

7 Conclusion

We introduce the first large-scale long form question answering dataset of open-ended queries with explanatory multi-sentence answers. We show that abstractive models generate coherent answers and are competitive with extractive models in human evaluation. Proposed models are far from human performance, in part due to the inability to exploit the long full web text. We hope ELI5 will inspire future work in all aspects of long-form QA, from the information extraction problem of obtaining information from long, multi-document input to generating more coherent and accurate paragraph-length answers.

References

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*, 5:135–146.

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

Eunsol Choi, He He, Mohit Iyyer, Mark Yatskar, Wen tau Yih, Yejin Choi, Percy Liang, and Luke Zettlemoyer. 2018. Quac: Question answering in context. In *EMNLP*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805.

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

Angela Fan, David Grangier, and Michael Auli. 2018. Controllable abstractive summarization. In *ACL Workshop on Neural Machine Translation and Generation*.

Jonas Gehring, Michael Auli, David Grangier, Denis Yarats, and Yann N Dauphin. 2017. Convolutional Sequence to Sequence Learning. In *Proc. of ICML*.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017. Billion-scale similarity search with gpus. *CoRR*, abs/1702.08734.

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

Tomas Kocisky, Jonathan Schwarz, Phil Blunsom, Chris Dyer, Karl Moritz Hermann, Gabor Melis, and Edward Grefenstette. 2018. The narrativeqa reading comprehension challenge. *TACL*.

Chin-Yew Lin. 2004. Rouge: a package for automatic evaluation of summaries. In *ACL Workshop on Text Summarization Branches Out*.

Peter J. Liu, Mohammad Saleh, Etienne Pot, Ben Goodrich, Ryan Sepassi, Lukasz Kaiser, and Noam Shazeer. 2018. Generating wikipedia by summarizing long sequences. In *ICLR*.

Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. Ms marco: A human generated machine reading comprehension dataset. *CoRR*.

Myle Ott, Sergey Edunov, David Grangier, and Michael Auli. 2018. Scaling neural machine translation. In *WMT*, pages 1–9. Association for Computational Linguistics.

Romain Paulus, Caiming Xiong, and Richard Socher. 2017. A deep reinforced model for abstractive summarization. *arXiv preprint arXiv:1705.04304*.

Alec Radford, Karthik Narasimhan, Tim Salimans, and Ilya Sutskever. 2018. Improving language understanding by generative pre-training.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for squad. In *ACL*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *EMNLP*.

Siva Reddy, Danqi Chen, and Christopher D Manning. 2018. Coqa: A conversational question answering challenge. *arXiv preprint arXiv:1808.07042*.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Neural machine translation of rare words with subword units. In *ACL*.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *ICLR*.

Anastasios Tombros and Mark Sanderson. 1998. Advantages of query biased summaries in information retrieval. In *SIGIR*.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. 2017. Newsqa: A machine comprehension dataset. In *ACL Workshop on Representation Learning for NLP*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Łukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. *NIPS*.

Ellen M. Voorhees. 2003. Overview of the TREC 2003 question answering track. In *TREC*.

Dirk Weissenborn, Georg Wiese, and Laura Seiffe. 2017. Making neural qa as simple as possible but not simpler. In *CoNLL*.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.