11747 Project: Question Answering with Multi-hop refinement

Senyu Tong[*] **Simran Kaur** [*] **Yansheng Cao** [*]
{senyut, skaur, yanshenc}@andrew.cmu.edu

1 Dataset and Task

In this project our team want to explore the task of open-domain question answering. We choose to work on Natural Questions(NQ) corpus (Kwiatkowski et al., 2019) created by Google AI Research using questions from real users and the NatQA (Khashabi et al., 2020) dataset constructed from it.

Each sample of NQ dataset is a question, an answer and an associated Wikipedia page context triplet. The task is to output a long answer (which is typically a paragraph) and a short answer (which is an entity) given the input. The common approach is to regard NQ as an Extractive QA(EX) dataset (Khashabi et al., 2020) which requires models to extract the answer as a substring from the context. NatQA, a variance of NQ, is an Abstractive QA (AB) dataset proposed by (Khashabi et al., 2020). NatQA only uses the question and short answer pairs in NQ, and with the document context eliminated, each sample is augmented with an additional context paragraph output by a DPR retriever (Karpukhin et al., 2020). Instead of predicting the answer span, in many cases the proposed answers are not the substrings of the retrieved context.

Given the Wikipedia page or the most relevant context, our task is a text-based QA instead of a knowledge-based one. Hence, the main challenge stems from how to take in large amounts of information, locate salient paragraphs for the long answer and the short answer, if they exist, and yield a probable answer even if it is not explicitly mentioned. Our improvement should be agnostic to whether the model predicts the answer span, as in EX datasets, or generates the answer itself, as in AB.

Furthermore, we hope to investigate whether adding multi-hop reasoning using the context paragraph + the question query can improve the performance of state of the art question answering systems/model framework.

2 Literature Review

From early Lunar systems to the IBM Watson jeopardy QA master, QA system have been an integral part of NLP research. With the advance of deep learning, large scale close-domain, and open-domain dataset, new QA systems have achieved comparable performance to human annotators on a variety of benchmarks. Moreover, there has been a drastic increase in works investigating new MRC-QA benchmark datasets and neural-network models, along with End-to-End architectures.

In the following section, we aim to provide a summary of some of the major machine reading comprehension dataset for Question Answering and discuss the characteristics and limitation of these datasets, along with the recent advances for QA systems and neural-network model architectures.

2.1 MRC and QA task type

There are four primary machine reading task: multiple choice, span prediction, cloze style, and open-domain answer.

Multiple-choice Question For multiple choice style questions, the QA system needs to select the correct answer to a question from a set of answers given a paragraph or document context.

Span Prediction For span prediction tasks, the QA system needs to select the correct beginning and end of the answer from the given context, which is equivalent of finding the start and end token of a given context.

Cloze-Style For cloze-style tasks, the QA system needs to obtain the most suitable phrases that can

[*] Everyone Contributed Equally – Alphabetical order

”fill in the blank” according to the context.

Open-answer and other categorization For open-answer tasks, the answer to a question is unrestricted and is not limited to particular span. Since this type of task is relatively coarse, (Zeng et al., 2020) demonstrates efforts to create new classification of the categories into finer subdivisions.

2.2 Dataset

In this section, we aim to provide a brief overview of the major public datasets and analyze their strengths, weaknesses, and key features.

CNN Daily News and Mail Dataset The CNN Daily and News dataset is a cloze-style machine reading comprehension dataset obtained from CNN Daily and News articles. The questions are created by replacing entities from bullet points summarizing one or several aspects of the article. The question answering systems are designed to infer the missing entity in the bullet point based on the content of the corresponding article. The CNN dataset contains 90,266 documents and 380,298 questions, while the Daily mail dataset contains 196,961 documents and 879,450 questions. Since its release in 2015, the dataset has been prevalent in the question answering/reading comprehension task. (Chen et al., 2016).

SQuaD The SQuaD dataset (Rajpurkar et al., 2016) is one of the most widely used dataset for machine reading comprehension. It contains more than 100,000 questions, and the answer for each question is a segment of text from a 100 - 150 word context paragraph from Wikipedia. This main feature of SQuAD is also a potential limitation, as a wide range of questions in the natural language world cannot be answered in this format. Regardless, SQuaD 1.1 remains the most popular MRC and QA dataset and the current state of the art models have surpassed the performance of human annotators.

TriviaQA The TriviaQA dataset (Joshi et al., 2017) contains more than 900 thousand question-answer pairs from 662 thousands documents collected from Wikipedia and web data. The TriviaQA is a challenging dataset to test QA system, as the questions are complex, contains combinatorial reasoning, and may even require cross sentence reasoning. TriviaQA poses one main challenge: it takes a significant amount of computational resources and time to preprocess and train large scale systems.

MS Marco MS MARCO dataset (Bajaj et al., 2018), released in 2016, is primarily used for document and passage level answering. The dataset consists of 1,010,916 questions and answers, which are generated from anonymous Bing’s search queries. The MS MARCO dataset can be used for multiple tasks, such as identifying questions that were unanswerable and ranking the passages retrieved given a question query. This dataset is not well suited for semantic search.

Hotpot QA Unlike MS MARCO, the HotpotQA dataset (Yang et al., 2018) is a multi-step reasoning dataset that aims to enable researchers and engineers to build models that perform complex, multi-hop reasoning and provide explanations for answers. The questions require the system to read and reason over multiple documents to find the answer and the questions are not subject to any pre-existing knowledge base. Another key feature of the HotpotQA dataset is that the data provides sentence-level supporting facts required for multi-step reasoning.

Natural Questions The Natural questions corpus (Kwiatkowski et al., 2019) released by Google is 2019 is an open-domain QA dataset. Unlike traditional close-book MRC tasks, open-domain QA tasks make no assumption about a given passage for context. The QA system has to deal with a large collection of documents and the goal is to return the answer for any open-domain questions without knowing the location of the answer beforehand. This poses more challenges and use-cases for QA systems that have strong performance on open-domain QA dataset. The NQ dataset is featured in NeurIPS 2020 question answering challenge https://efficientqa.github.io.

2.3 Systems and Architectures

In this section, we aim to provide a brief overview of relevant QA systems and architectures, and analyze their strengths, weaknesses, and key features.

BERT Motivated by the fact that language model pre-training has successfully improved performance on many natural language processing tasks, (Devlin et al., 2018) built the Bidirectional Encoder Representations from Transformers (BERT) model.

While the BERT model performs well on many language tasks, it suffers from the memory system and performs poorly on capturing long range dependency. While the history observation in the language model is important, the computational cost of attending to every time-step and the storage cost of preserving this large memory are quite expensive. (Rae et al., 2019) proposed a compressive transformer to transfer past observations into a smaller set of compressed representations, which is suitable for one-direction modeling. (Zaheer et al., 2020) extended the work into bidirectional contexts.

Reformer The Reformer (Kitaev et al., 2020) addresses some computational drawbacks of transformers like BERT. The reformer provide significant memory-efficiency gains and faster performance on longer sequences, without compromising model performance.

However, (Katharopoulos et al., 2020) noted that reformers cannot be used for decoding tasks where the keys need to be different from the queries. Since reformers use local-sensitivity hashing for efficiency gains, reformers constrain the keys and queries for the attention to be identical.

Routing Transformer Like the Reformer, the Routing Transformer also addresses computational drawbacks of transformers like BERT. The Routing Transformer (Roy et al., 2020) combines efficient content-based sparse attention with classic local attention to achieve model flexibility and efficiency gains.

While the Reformer uses local-sensitivity hashing to infer content based sparsity patterns for attention, the Routing Transformers uses spherical k-means to do so, which is known to outperform local-sensitivity hashing in approximating Maximum Inner Product Search (MIPS).

UnifiedQA Pretraining on in-domain unlabeled data is an effective transfer learning technique in NLP. By using a unified framework that converts all text processing problem as a “text-to-text” problem, (Raffel et al., 2019) found that pre-training on in-domain unlabeled data can improve performance on QA tasks. They fed a modified transformer the question and context, and asked to generate the answer token-by-token. This contrasts the outputs of BERT-style models, which is either a class label or span of the input. (Raffel et al., 2019) found that pretraining on data from Wikipedia produced significant gains on the SQuAD dataset.

(Izacard and Grave, 2020) initialized models with pre-trained T5 models (Raffel et al., 2019) and improved the performance of QA by increasing the number of retrieved passages based on the Natural Questions and TriviaQA open benchmarks.

Similarly, (Li et al., 2020) proposed reframing Named Entity Recognition (NER) tasks as MRC tasks instead of as a sequence labeling task, and achieved SOTA performance on nested NER datasets. This unified framework was capable of handling both flat and nested NER tasks, in addition to encoding prior knowledge about the entity category in the query.

Multi-task objective In the long-form question answering area, (Fan et al., 2020) proposed an abstractive model trained with a multi-task objective, which performs better than conventional Seq2Seq, language modeling, and a strong extractive baseline. However, the model’s performance is still far from human performance.

Attention-Based Neural Matching Model (Yang et al., 2016) proposed an attention-based neural matching model for ranking short answer text, which combines different matching signals and incorporates question-term importance learning using a value-shared weighting scheme. The proposed model is simpler and yields better performance than the combination of conventional neural networks and additional features (word overlap or BM25 scores) based on the benchmark TREC QA data.

BiDAF BiDAF (Seo et al., 2018) uses character-level, word-level, and contextual embeddings to represent the context at different levels of granularity and uses bi-directional attention flow to obtain a query-aware context representation.

Unlike previous attempts to apply attention to QA, BiDAF uses bidirectional attention flow: query-to-context and context-to-query. Instead of using attention layers to summarize the context paragraph into a fixed-size vector, BiDAF takes advantage of the attention flow to the modeling (RNN) layer to reduce the information loss caused by early summarization. Furthermore, the attention at each time step is a function of only the query and the context paragraph at the current time step, making BiDAF a memory-less attention mechanism.

(Seo et al., 2018) demonstrated that BiDAF achieves SOTA results in SQuAD and CNN/DailyMail cloze test.

BiDAF + GloVE + ELMo (Peters et al., 2018) observed further improvements on SQuAD by adding ELMo to the BiDAF architecture. BiDAF already uses pretrained word vectors, GloVE, in the word embedding layer to obtain the fixed word embeddings. (Peters et al., 2018) incorporated ELMo, pretrained word embeddings that are function of all of the internal layers of the BiDAF. In particular, the ELMo embeddings were concatenated with the input to the character-level embeddings and with the representations outputted by the contextual embeddings.

ETC: Encoding Long and Structured Inputs in Transformers Most variants of the standard Transformer limit inputs to n = 512 tokens due to attention limitations and rarely focus on structured inputs. ETC (Ainslie et al., 2020) can scale attention to longer inputs and encode structured inputs by using a novel global-local attention mechanism as well as relative positional encodings and Contrastive Predictive Coding (CPC) pretraining. For long answer NQ, (Ainslie et al., 2020) observed that increasing input length and lifting weights from existing RobERTA models improved ETC dev performance compared to baseline models (BERT-base, BERT-large, and RikiNet).

3 Baselines

We select BERT-joint (Alberti et al., 2019), DPR (Karpukhin et al., 2020) + BART (Lewis et al., 2019), and DPR + UnifiedQA (Khashabi et al., 2020) as our baselines.

3.1 BERT-joint

Motivation Following (Alberti et al., 2019), we use a BERT based model as our first baseline. BERT (Devlin et al., 2018) is empirically powerful on machine comprehension tasks, and we are curious if it could scale well on a harder open-domain question answering task.

Setting and Dataset We use Google's official codebase and experimental setting to reproduce the results. Instead of using a pipeline approach, in (Alberti et al., 2019) both short and long answers are predicted in a single model. Also, the "[CLS]" token is used to predict null instances at training time and rank spans at inference time. We also use the pre-processed NQ-train and NQ-dev dataset realeased by Google. Each of 7,830 sample is tokenized following (Devlin et al., 2018), and special markup tokens are used to indicate different paragraphs and tables in the document. Each training instance is defined to be a four-tuple (c, s, e, t) standing for context of wordpiece ids, indices pointing to the start and end of the answer span, and the annotated answer type. Following (Alberti et al., 2019), we initialize the training from a BERT-large model trained on SQuAD2.0 and then finetune it on NQ for 1 epoch with learning rate of 3e-5.

Evaluation Metric During inference time, the prediction is outputted as index spans. We use the official evaluation script released by Google. F1, precision and recall scores are calculated. For the short answer, both the answer span and the yes/no answer are taken into account. If more than one annotator marked the short as answerable or a yes/no question, then the prediction must match one of the given span of yes/no label. All span comparisons are exact.

Result As shown in 1, following (Alberti et al., 2019), we successfully reproduce their results on the paper. We borrow results in previous baselines DocumentQA (Clark and Gardner, 2017), DecAtt (Parikh et al., 2016), and Document Reader (Chen et al., 2017) and for the performance of human annotators.

3.2 DPR + BART

Motivation We select another Transformer-based pretrained model, BART (Lewis et al., 2019), as one of our baselines for two reasons. First, BART is a pretrained denoising autoencoder. We believe that due to the nature of Natural Question dataset, Wikipedia pages tend to be long and contain extraneous information that may make the machine comprehension problem harder, such as HTML tokens and charts. Hence, we need the model to be flexible with noise. Second, BART is a strong model that is particularly effective for comprehension tasks (Lewis et al., 2019) and matches the performance of RoBERTa.

Setting and Dataset We use the code-base released by Allenai. To be consistent with (Lewis et al., 2019), our model is not directly fine-tuned on NQ raw data, but on NQ-dev with DPR parameters, i.e. the NatQA dataset as introduced in section 1. Though we omit the original Wikipedia document, each question-answer pair is augmented with the most relevant paragraphs retrieved by DPR (Karpukhin et al., 2020) retriever. Due to computa-

	Long Answer Dev			Short Answer Dev		
	P	R	F1	P	R	F1
DocumentQA	47.5	44.7	46.1	38.6	33.2	35.7
DecAtt + DocReader	52.7	57.0	54.8	34.3	28.9	31.4
BERT$_{\textbf{joint}}$	**61.3**	**68.4**	**64.7**	**59.5**	**47.3**	**52.7**
Single Human	80.4	67.6	73.4	63.4	52.6	57.5
Super-annotator	90.0	84.6	87.2	79.1	72.6	75.7

Table 1: BERT-joint results on NQ dataset

	NatQA
DPR + BART$_{\text{base}}$	30.29
DPR + UnifiedQA	33.21

Table 2: The performance (Accuracy) of baselines on NQ-devset

tional limits, we could not finetune a BART-Large model, as it exceeds 16G GPU memory. Therefore, we finetuned Facebook's pretrained BART-base model for two epochs. It takes around 5 hours for finetuning on a single Tesla V-100 GPU.

Evaluation Metric and Result During inference, we output short answers. Following the original experiment, we compare the given answer and predicted output, both in lowercase, ignoring spacing and special tokens such as quotation marks and foreign characters. Note that this evaluation metric is different from the official NQ standard, where the inference output are the start/end indexes of answers instead of actual word sequences. We avoid de-tokenization for string comparison by comparing the spans directly. We also omit yes/no tags and unanswerable tags from the original dataset. We achieve 30.29% accuracy. The average edit distance for wrong answers is 11.5.

Error Analysis

- Out of 10,693 samples, 3,387 answers are not wrapped in the retrieved paragraphs. This is consistent with the performance of DPR as in (Karpukhin et al., 2020), among them, 3,292 got wrong predictions. We believe that, since we fail to cover the answer span in the retrieved context, the answer prediction is largely negatively influenced.

- the model fails to understand questions that require multiple entities as answers, or more generally, the model fails to understand "numbers." For example, the question *"5 types of control that could be programmed on a gui?"* expects 5 outputs *"spinner; drop-down list; slider; button; list box"*, but the model could only output one, *"command-line"*. While it is easy for the model to output multiple answers according to ranking of scores, it is harder to output multiple answers according linguistic reasoning.

- As in figure 1, the lengths of most questions are 7 to 12 words. The results are 28.7% and 32.1% for questions with 8 and 13 words respectively. But there is no clear relationship between question length and accuracy .

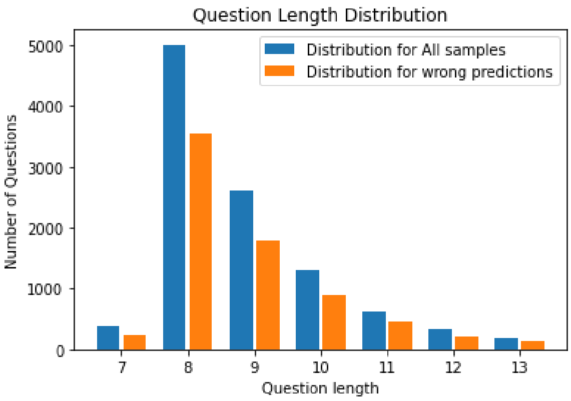

Figure 1: Question Length Distribution

- There are 4478 questions start with *"who"*. Following that are *"when", "what", "where"* and *"how"* with 1877, 1378, 889, and 492 samples respectively. As in Figure 2, the model performs worst for questions start with *"what"*.

3.3 DPR + UnifiedQA

Motivation UnifieQA (Khashabi et al., 2020) is a single pre-trained QA model that performs well

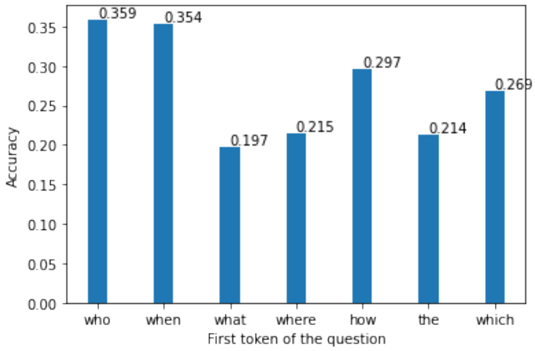

Figure 2: Accuracy w.r.t the question starts

across multiple datasets spanning both Extractive (EX) and Abstractive (AB) formats. Instead of specializing for certain dataset and quest types, UnifiedQA seek to teach reasoning abilities that are not governed by the format. We want to explore how changing the QA format from predicting the indices span to abstraction influence the performance, and also if our proposed improvements are agnostic to the formats.

Setting and Dataset As in the previous subsection, we use the identical code base for UnifiedQA. We finetune a UnifiedQA model with BART-large as backbone on NatQA dataset for three epochs each with 10,000 steps. We use the checkpoint realeased by Allenai as our initial model weights.

Due to the computational limit of 16G GPU, we could not follow the paper drawing 120 samples per training batch. Instead, we could only set batch size 4 and the finetuning process is extremely slow.

Evaluation Metric and Result The evaluation metric is identical to previous subsection. When a question accepts multiple answers, the evaluation script views each question-answer pair as an independent sample and therefore even the model predicts one of the answers, all other pairs are regarded to be wrong. Also, even after normalizing the tokenized true answers, we find many special tokens for foreign characters exist, like *"\xa2"*, *"\xc3"*. The model can not output those characters and thus it will fail on these cases. Note that, every sample in NatQA dev set is answerable, with short answer fewer than 5 tokens, and there is no yes/no question. Hence we suppose the comprehension problem is harder than the original NQ dataset and therefore we expect the precision discrepancy. The dev set we use has 10,693 samples, unlike the published result which inference on only 3.6k instances. We got 33.21% accuracy after finetuning for 3 epochs. We did not observe any significant difference between the error pattern of finetuning UnifiedQA and BART.

4 Approach and Future Plan

We aim examine the robustness of unifiedQA and BERT_Joint to position bias by separating the Natural Questions dataset into samples with answers in the first sentence of a context vs samples with answers in other parts of a context and Research have shown that for LM such as BERT pretrained on SQuaD with answers that can be found in the first sentence of the context, the correlation coefficient defined by cosine similarity is much higher in the synthetic dataset, which is not desirable although this exploitation may help boost performance on the given dataset because real-application scenarios do not necessarily follow this distribution.

Another interesting approach that we hope to try to incorporate Multi-hop reasoning in the question and context generation phase, instead of working with the exact question, we want to explore the possibility of adding multi-hop reasoning to iterate between finding/reading the context and generating a refined search/question query.

References

Joshua Ainslie, Santiago Ontanon, Chris Alberti, Vaclav Cvicek, Zachary Fisher, Philip Pham, Anirudh Ravula, Sumit Sanghai, Qifan Wang, and Li Yang. 2020. Etc: Encoding long and structured inputs in transformers.

Chris Alberti, Kenton Lee, and Michael Collins. 2019. A bert baseline for the natural questions.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, Mir Rosenberg, Xia Song, Alina Stoica, Saurabh Tiwary, and Tong Wang. 2018. Ms marco: A human generated machine reading comprehension dataset.

Danqi Chen, Jason Bolton, and Christopher D. Manning. 2016. A thorough examination of the CNN/daily mail reading comprehension task. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2358–2367, Berlin, Germany. Association for Computational Linguistics.

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

Christopher Clark and Matt Gardner. 2017. Simple and effective multi-paragraph reading comprehension.

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

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2020. Eli5: Long form question answering. *arXiv preprint arXiv:1907.09190v1*.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282v2*.

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

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen tau Yih. 2020. Dense passage retrieval for open-domain question answering.

Angelos Katharopoulos, Apoorv Vyas, Nikolaos Pappas, and François Fleuret. 2020. Transformers are rnns: Fast autoregressive transformers with linear attention.

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.

Nikita Kitaev, Lukasz Kaiser, and Anselm Levskaya. 2020. Reformer: The efficient transformer. In *International Conference on Learning Representations*.

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

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension.

Xiaoya Li, J. Feng, Yuxian Meng, Qinghong Han, F. Wu, and J. Li. 2020. A unified mrc framework for named entity recognition. In *ACL*.

Ankur Parikh, Oscar Täckström, Dipanjan Das, and Jakob Uszkoreit. 2016. A decomposable attention model for natural language inference. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, pages 2249–2255, Austin, Texas. Association for Computational Linguistics.

Matthew Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 2227–2237, New Orleans, Louisiana. Association for Computational Linguistics.

Jack W Rae, Anna Potapenko, Siddhant M Jayakumar, and Timothy P Lillicrap. 2019. Compressive transformers for long-range sequence modelling. *arXiv preprint arXiv:1911.05507*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J Liu. 2019. Exploring the limits of transfer learning with a unified text-to-text transformer. *arXiv preprint arXiv:1910.10683*.

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

Aurko Roy, Mohammad Saffar, Ashish Vaswani, and David Grangier. 2020. Efficient content-based sparse attention with routing transformers.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2018. Bidirectional attention flow for machine comprehension.

Liu Yang, Qingyao Ai, Jiafeng Guo, and W. Bruce Croft. 2016. anmm: Ranking short answer texts with attention-based neural matching model. *arXiv preprint arXiv:1801.01641v2*.

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.

Manzil Zaheer, Guru Guruganesh, Avinava Dubey, Joshua Ainslie, Chris Alberti, Santiago Ontanon, Philip Pham, Anirudh Ravula, Qifan Wang, Li Yang, et al. 2020. Big bird: Transformers for longer sequences. *arXiv preprint arXiv:2007.14062*.

Changchang Zeng, Shaobo Li, Qin Li, Jie Hu, and Jianjun Hu. 2020. A survey on machine reading comprehension: Tasks, evaluation metrics and benchmark datasets.