Question Answering on SQuAD2.0 and QuAC

Siyuan Ding
University of Pennsylvania
siyuand4@upenn.edu

Renisa Pati
University of Pennsylvania
renisa@seas.upenn.edu

Yinuo Xie
University of Pennsylvania
yinuoxie@upenn.edu

Abstract

Question answering (QA) is a task in natural language processing (NLP) that involves automatically answering questions posed in natural language. The SQuAD and QuAC datasets are two widely used benchmarks for evaluating the performance of QA systems. In this paper, we evaluate different models' performance for QA on the SQuAD and QuAC datasets. We also extended to build an "open-domain" QA system. However, unlike normal open-domain QA, which uses entire Wikipedia pages as a resource, we only considered the passages in SQuAD and QuAC datasets as our "Wikipedia." Additionally, we provided an in-depth analysis of the strengths and limitations of each model evaluated.

1 Introduction

Natural language processing (NLP) is a field of artificial intelligence (AI) that focuses on enabling computers to understand, interpret, and generate human language. One important application of NLP is question answering (QA), which involves using a computer system to generate a natural language response to a user's question. QA systems can be used to answer a wide range of questions, including those related to general knowledge, technical information, or specific domains such as health or finance. QA system usually follows the steps below.

1. The first step of the QA system is to process the question and passage with a tokenizer.

2. Then, the embeddings will be fed into a machine learning or deep learning model to predict the answers' start and end positions in the passage.

3. The actual answer is just the span of words in the passage from the predicted start and end position. If the index of the start position is larger than the end position, then no answer is generated

In this paper, we divided our work into three parts.

1.1 Simple Baseline

We initially built a simple baseline, randomly selecting the start and end positions from the context to generate an answer to the user's question.

1.2 BERT Model

In recent years, there has been significant progress in the field of QA due to the development of powerful deep learning models, such as BERT (Bidirectional Encoder Representations from Transformers). So, we made our first extension to explore using BERT for QA (Figure 1). We then evaluated the performance of various pre-trained BERT models on SQuAD2.0 and QuAC datasets.

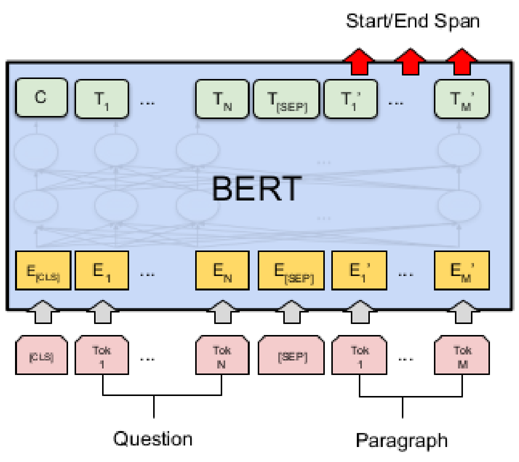


Figure 1: QA System with BERT(N, 2020)

1.3 Document Retrieval + BERT Model

Recently, open-domain QA has been trending in the field of QA that allows the machine answers to a user's question, where the domain of the question is not restricted or predetermined. The system is designed to handle a wide range of questions without requiring users to specify the source or type of information they seek. This requires the QA system to have a broad understanding of the world and to be able to identify and retrieve relevant information from a wide range of sources.

Thus, we made our second extension to explore how the open-domain QA pipeline works. Nevertheless, with limited computing resources, we had to limit our domain to the SQuAD2.0 and QuAC datasets. Figure 2 shows an example of the Open-Domain QA pipeline. In this pipeline, the QA system first searches for and retrieves the top k relevant passages according to the question (Document Retrieval). Then the BERT model will be used to select and generate a response based on the retrieved information (Document Reader). In Document Retrieval, we implemented two algorithms, lexical search and passage ranking. In Document Reader, we reused the model evaluated in section 1.2. Finally, we combined these two parts and assessed their overall performance on SQuAD2.0 and QuAC dataset.

2 Literature Review

Jacob Devlin et al.(Devlin et al., 2019) improve the fine-tuning-based approaches for question-answering by proposing BERT, Bidirectional Encoder Representations from Transformers. BERT addresses the limitation caused by unidirectionality by using a ”masked language model” (MLM) pre-training objective. The MLM objective differs from the left-to-right pre-training in that it facilitates the representation to fuse the left and the proper context, which allows the training of a deep bidirectional Transformer. They also introduce the ”next sentence prediction” task that jointly pre-trains text-pair representations. To make BERT handle various downstream tasks, their input representation unambiguously represents both a single sentence and a pair of sentences. Its input representation is constructed for a given token by summing the corresponding token, segment, and position embeddings. To train a deep bidirectional representation, the mask some percentage of the input tokens at random, followed by prediction of the masked tokens. On SQuAD, they treat questions that do not have an answer as having an answer span with start and end at the [CLS] token. The probability space for the start and end answer span positions is extended to include the position of the [CLS] token. They consider 11 NLP tasks and use the GLUE benchmark for evaluation of the BERT pre-trained model and present how BERT outperforms other task-specific architectures with an average score of 82.1.

Choi et al.(Choi et al., 2018) present QuAC dataset, a Question Answering in Context dataset containing 14K information-seeking QA dialogues. QuAC is more challenging in a way where its questions are often more open-ended, unanswerable, or only meaningful within the dialog context. They emphasize the role of context in answering an open-ended question and show how the answer's location within the text is influenced by the number of questions asked previously. They use a Pre-trained Inferset baseline based on cosine similarity, Feature-rich logistic regression, which uses n-gram overlap, bias features, and contextual features. Further, they re-implement a top-performing SQuAD model, BIDAF++, where bidirectional attention flow is augmented with self-attention and contextualized embeddings. Additionally, they modify the passage and question embedding processes to consider the dialog history. After experimentation with varied models, they observe that even their best model under-performs humans and achieves human equivalence on only 60% of questions and 5% of complete dialogues, which indicates that more sophisticated models are required to exploit the presence of context.

Rodrigo Nogueira et al (Nogueira and Cho, 2019). describe a re-implementation of BERT for query-based passage re-ranking. They state that the job of the re-ranker here is to estimate a score s_i of how relevant a candidate passage d_i is to a query q. They use BERT as their re-ranker, feed the query as sentence A, the passage text as sentence B, and truncate the query to have at most 64 tokens. They set the maximum length of their concatenated sequence to 512 tokens. BERT-large model is used as their binary classification model where the [CLS] vector acts as input to a single-layer neural network to obtain the probability of the passage being relevant. The probability for each passage is computed independently, and the ranking of passages of based on this probability. They begin with a pre-trained BERT model and fine-tune it to cater to their re-ranking task using the cross-entropy loss. They train and evaluate their model

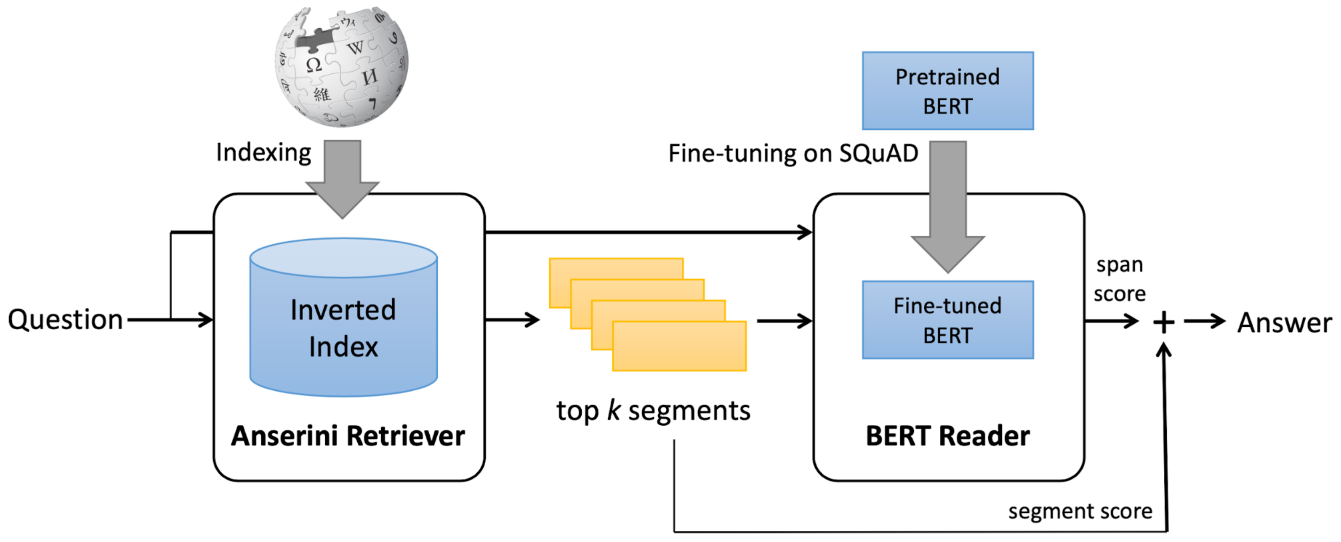


Figure 2: Example of Open-Domain QA Pipeline(Weng, 2020)

on MS MARCO and TREC-CAR, two passage-ranking datasets, and observe that the pre-trained models need few training examples to achieve a good score.

3 Experimental Design

3.1 Dataset

For our Question Answering task, we are using the SQuAD2.0 dataset, which combines the 100,000 questions in SQuAD1.1 with over 50,000 unanswerable questions written in an adversarial manner by crowd-workers to look similar to answerable ones. Both the train and the validation files contain data in the below format (Table 1). Table 2 displays an example of the SQuAD2.0 dataset, and Table 3 shows the distribution of the dataset.

title	The title of the Wikipedia article. (String)
context	The full text of the Wikipedia article. (String)
question	The question that the model will be asked. (String)
answers	The answer to the question. (String)

Table 1: Format of SQuAD2.0 Dataset

Instance of the data:

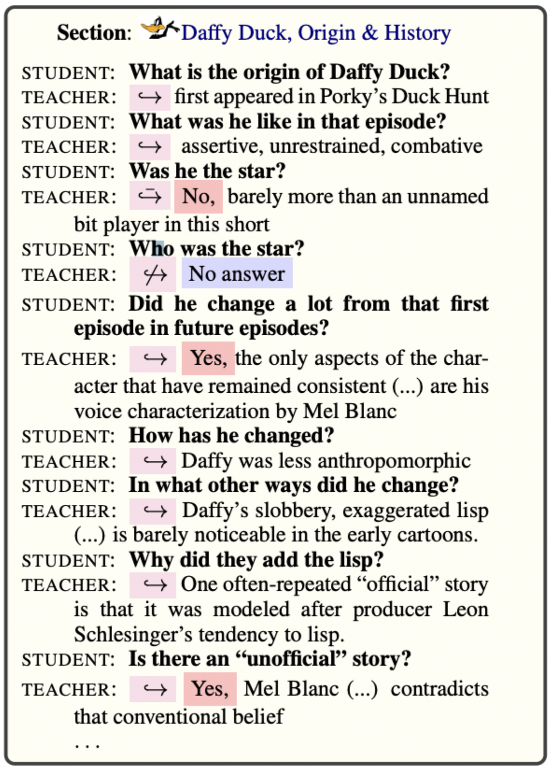
As our extension, we also worked on the QuAC, Question Answering in Context dataset, which is meant for modeling, understanding, and participating in the information-seeking dialog. Data instances here consist of an interactive dialog between two crowd workers: (1) a student who poses a sequence of freeform questions to learn as much as possible about a hidden Wikipedia text, and (2) a teacher who answers the questions by providing short excerpts (spans) from the text. QuAC's questions are often more open-ended, unanswerable, or only meaningful within the dialog context (Choi et al., 2018). Figure 3 displays an example of QuAC dataset. Our reasoning behind experimenting with the models on QuAC was that QuAC shares many principles with SQuAD2.0, such as span-based evaluation and unanswerable questions. However, it incorporates a new dialog component

title	Normans
context	The Normans (Norman: Nourmands; French: Normands; Latin: Normanni) were the people who in the 10th a...
question	In what country is Normandy located?
answers	{'text': array(['France', 'France', 'France', 'France'], dtype=object), 'answer_start': array([159, ...

Table 2: Example of SQuAD2.0 Dataset

	unique rows	title	context	questions
Train	130319	442	19029	130217
Validation	11873	n/a	1204	11864

Table 3: Distribution of SQuAD2.0 Dataset


Section: Daffy Duck, Origin & History

STUDENT: **What is the origin of Daffy Duck?**
TEACHER: ↪ first appeared in Porky's Duck Hunt
STUDENT: **What was he like in that episode?**
TEACHER: ↪ assertive, unrestrained, combative
STUDENT: **Was he the star?**
TEACHER: ↪ No, barely more than an unnamed bit player in this short
STUDENT: **Who was the star?**
TEACHER: ↛ No answer
STUDENT: **Did he change a lot from that first episode in future episodes?**
TEACHER: ↪ Yes, the only aspects of the character that have remained consistent (...) are his voice characterization by Mel Blanc
STUDENT: **How has he changed?**
TEACHER: ↪ Daffy was less anthropomorphic
STUDENT: **In what other ways did he change?**
TEACHER: ↪ Daffy's slobbery, exaggerated lisp (...) is barely noticeable in the early cartoons.
STUDENT: **Why did they add the lisp?**
TEACHER: ↪ One often-repeated "official" story is that it was modeled after producer Leon Schlesinger's tendency to lisp.
STUDENT: **Is there an "unofficial" story?**
TEACHER: ↪ Yes, Mel Blanc (...) contradicts that conventional belief
...

Figure 3: Example of QuAC (Choi et al., 2018)

that needs to be accommodated.

3.2 Evaluation Metric

To evaluate the retriever, we retrieve the top 5 relevant passages to the questions in the SQuAD2.0 and QuAC dataset and calculate the percentage that the retrieved passages contain the actual passage.

$$score = \frac{\text{\# of selected passages with real passage}}{\text{\# of all questions}}$$

To evaluate the reader, we use the standard SQuAD2.0 performance metrics: Exact Match (EM) score and F1 score. For our project, we focus on the EM and F1 scores with respect to the dev set.

- Exact Match: A binary measure of whether the system output matches one of the ground truth answer exactly, excluding the unanswerable questions.

- F1: Harmonic mean of precision and recall, where

$$\text{precision} = \frac{\text{true positives}}{\text{true positives + false positives}}$$
$$\text{recall} = \frac{\text{true positives}}{\text{false negatives + true positives}}$$

and

$$\text{F1 score} = \frac{2 \times \text{prediction} \times \text{recall}}{\text{precision + recall}}$$

The F1 score will be max of all scores of the golden answers.

In evaluation, precision measures to what extent the predicted span is contained in the ground truth span, and recall measures to what extent the ground truth span is contained in the predicted span.

Our experiment design follows the same format as the paper. We will evaluate the performance of simple baseline and various BERT models on SQuAD2. 0 and QuAC dataset. Then we will make an extension to the Document Retriever + BERT model.

3.3 Simple Baseline

We initially built a simple baseline, randomly selecting the start and end positions from the context to generate an answer to the user's question.

Dataset	Train EM/F1	Dev EM/F1
SQuAD2.0	0.000/0.029	0.000/0.031
QuAC	0.000/0.043	0.000/0.065

Table 4: Performance of Simple Basline

4 Experimental Results

4.1 BERT Model

We made the first extension to apply BERT models. In BERT, the input query and passage pair are tokenized using WordPiece embeddings and then packed into a sequence. A special classification token embedding, denoted as [CLS], is always added to the beginning of the sequence and is used as an aggregated representation of the entire sequence. A separation token, denoted as [SEP], is added between the query and passage at the end of the passage to separate the two. A pair of learned sentence segment embeddings is used to further differentiate the query and passage. Each query token, including [CLS], is paired with the segment A embedding, while each passage token is paired with the segment B embedding. Finally, position embedding is added to each token. The final input to BERT is the sum of the text, segment, and position embeddings. The input then goes through layers of bidirectional Transformers to produce hidden states that can be used as contextual representations of the input query-passage pair (Devlin et al., 2019).

Initially, we used the BERT("bert-base-uncased") model and attempted to improve its performance by fine-tuning it on the SQuAD 2.0 dataset. We used the cross-entropy loss as the criterion and Adam gradient descent as the optimizer. However, due to our limited computing resources on Google Colab, we were unable to significantly improve the model's performance in a reasonable amount of time. Therefore, after carefully searching on Hugging Face, we decided to use some pre-trained and fine-tuned BERT models on SQuAD 2.0 and/or QuAC. The results of these models are displayed in the Table 5.

In Table 5, we can observe that the Bert-base-uncased model performs poorly because it is not fine-tuned on any dataset. However, when we fine-tuned the pre-trained BERT model, we achieved a relatively high EM and F1 score. The Bert-large-finetuned-squad model, as cited in (Devlin et al., 2019), is fine-tuned on the SQuAD2.0 dataset and performs significantly better on that dataset, achieving the stated EM and F1 scores in the original paper. However, without fine-tuning on the QuAC dataset, the model does not perform as well on that dataset. Therefore, we tried two other models that are fine-tuned on both SQuAD2.0 and QuAC datasets. While their performance on SQuAD2.0 decreases, their EM and F1 scores on QuAC improve compared to the Bert-large-finetuned-squad model.

4.2 Document Retriever + BERT Model

Our second extension was to combine the Document Retriever and use the BERT model as our Document Reader to explore how open-domain QA works. We implemented two document retrievers: lexical search and passage ranking. For lexical search, we applied the BM250Kapi package in Rank-BM25. Rank-BM25 uses a combination of term frequency and inverse document frequency (TF-IDF) to calculate the relevance score for each document. BM250Kapi is an extension of Rank-BM25 that considers the term frequency, document length, and the position of the terms within the document. For passage ranking, we encoded all passages into our vector space. Then we performed the semantic search, where the queries are encoded using the bi-encoder to find potentially relevant passages. Next, we scored all retrieved passages with the cross-encoder, sorted the results, and fetched the highest-scoring passages for answer extraction. The performance of two retriever methods is shown in Table 6.

Once we had obtained the highest-scoring passages, we applied the BERT model (section 3.4.1) to extract the answer. During prediction, we choose the best span from the start to end position generated by each pair of questions such that $P(\text{start}) \times P(\text{end})$ is maximized (Chen et al., 2017). Table 7 below displays their performance. Upon closer examination of the results in Table 7, we find that the performance of the combined model is roughly equal to the product of the retriever match rate and the reader model performance. This suggests that to improve the overall performance of an open-domain system, we can either focus on improving the effectiveness of the retriever algorithms or improving the performance of the reader model.

4.3 Error Analysis

For our fine tuned pretrained bert model, we tend to have unmatched answers whereas the model missed some further condition for the true answer leading to a relatively higher score for F1 than EM.
An instance of when it produced exact response:

- Question: *Downtown Burbank is an example of what kind of district?*
- Prediction: **business**
- True Answer: ['**business**', 'major business districts', 'major business']

For the example shown above we reached a True EM score and 1.0 F1 score.
An instance of when it produced partially correct/incorrect answer:

- Question: *What is the scenario called in which we don't change our greenhouse gas creation practices?*
- Prediction: **business as usual**
- True Answer: ['"**business as usual**" (BAU)', 'enhanced greenhouse effect', '"business as usual" (BAU)']

This is a very typical error example for our model where the algorithm successfully identifies the part which is highly correlated with the question query but missed the extension part.

	SQuAD2.0		QuAC	
Model	Train EM/F1	Dev EM/F1	Train EM/F1	Dev EM/F1
Bert-base-uncased	0.005/0.004	0.003/0.071	0.0/0.04	0.0/0.04
Bert-large-finetuned-squad (Devlin et al., 2019)	0.866/0.961	0.857/0.922	0.160/0.321	0.120/0.285
SciBERT-SQuAD-QuAC	0.321/0.556	0.249/0.518	0.301/0.425	0.279/0.399
Bert-finetuned-quac (Otegi et al., 2020)	0.237/0.424	0.201/0.382	0.212/0.412	0.180/0.374

Table 5: BERT Model Performance on SQuAD2.0 and QuAC

Retriever	Match Rate
Lexical Search	0.772
Passage Ranking	0.793

Table 6: Match Rate of Retriever methods

In this situation, we reached a False EM score and 0.86 F1 score.

On the other hand, our model performed some similar errors when training on QuAC however most of the errors exhibited completely mismatching.
An instance of when it produced similarly partially correct/incorrect answer:

- Question: *What did spacey say in response?*
- Prediction: **he would be seeking ”evaluation and treatment”**
- True Answer: [’On November 1, Spacey stated that **he would be seeking ”evaluation and treatment”** for his behavior.’ etc..]

In this situation, we reached a False EM score and 0.7 F1 score.
Despite from the portion of predictions that managed to match most of the true answers, most of the predictions generated on the QuAC dataset fails to reach that and only managed to match little of the true answers. An instance of when it produced mostly incorrect answer:

- Question: *Did he hold a political office?*
- Prediction: **sitting Labour MP, Bob Mellish, announced his retirement**
- True Answer: [’Tatchell was de**nounced** by party leader Michael Foot for allegedly supporting extra-parliamentary action against the Thatcher’, etc..]

From the above analyis we can drive a possible reason why our model tends to fail to capture most of the true answers when testing on QuAC. Our current model prefers questions with shorter answers such as questions in SQuAD2.0 rather than questions in QuAC, which tend to have long answers. Since most of the answers from SQuAD2.0 is short and concise, most of the error cases managed to obtain a high F1 score while the opposite situation is shown while using QuAC.

5 Conclusions

In conclusion, it can be stated that for QA systems to perform well on datasets like SQuAD2.0 and QuAC, several factors need to be addressed. From our experiments and analysis, we find that performance on QA tasks can be improved significantly through the below means.

- Larger and more diverse dataset: This can help the model learn more about the language and better understand the context in which questions are asked. We observed how Bert-large performed better since it had been trained with more parameters and on a larger dataset.
- Advanced NLP techniques: attention mechanisms, transformer models, and BERT-based models achieved superior results compared to simple naive baselines like random sequence or nearest neighbor.
- Combination of retrieval-based and generation-based approaches: Retrieval-based approaches involve selecting the most relevant information from a pre-defined set of documents. In contrast, generation-based approaches involve generating an answer from scratch. Combining these two approaches can often lead to better performance on question-answering tasks. Although this

	SQuAD2.0	QuAC
Model	Dev EM/F1	Dev EM/F1
Lexical + Bert-base-uncased	0.003/0.071	0.0/0.04
Passage + Bert-base-uncased	0.005/0.091	0.0/0.05
Lexical + Bert-large-finetuned-squad	0.614/0.695	0.092/0.219
Passage + Bert-large-finetuned-squad	0.677/0.729	0.094/0.285
Lexical + SciBERT-SQuAD-QuAC	0.191/0.399	0.214/0.307
Passage + SciBERT-SQuAD-QuAC	0.196/0.409	0.220/0.315
Lexical + Bert-finetuned-quac	0.155/0.294	0.139/0.288
Passage + Bert-finetuned-quac	0.159/0.302	0.142/0.296

Table 7: Document Retriever + BERT Model Performance on SQuAD2.0 and QuAC

technique did not work for us since we could not implement true open-domain QA due to computing constraints, it can be beneficial for many QA tasks.

- Fine-tuning of the model on a specific domain: Fine-tuning the model on a specific domain (e.g., medicine, law) can help it learn domain-specific language and concepts and improve its performance on question-answering tasks in that domain. This can help avoid out-of-context responses by the model, and the model will also be able to recognize unanswerable questions better.
- Abstain from answering: As tricky as this might sound, the model must not only answer questions when possible but also determine when the paragraph supports no answer and abstain from answering.

6 Acknowledgments

We would like to express our sincere gratitude to Professor Mark Yatskar for a great semester. His lectures were engaging, informative, and helped us to develop a deeper understanding in NLP. We also want to thank all of the teaching assistants for their help with homework and other course-related tasks. Their guidance and support were invaluable in helping us to succeed in this course. We would like to give a special thanks to our TA mentor Siyi Liu for his assistance with our final project. His insights and guidance were invaluable in helping us to complete our project successfully. We are grateful for his expertise and his willingness to help us out. Thank you all for your contributions to our education and for making this a memorable and rewarding semester.

References

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

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

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.

Nishanth N. 2020. *Building a Question Answering System with BERT*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. arXiv.

Arantxa Otegi, Jon Ander Campos, Gorka Azkune, Aitor Soroa, and Eneko Agirre. 2020. Automatic evaluation vs. user preference in neural textual QuestionAnswering over COVID-19 scientific literature. In *Proceedings of the 1st Workshop on NLP for COVID-19 (Part 2) at EMNLP 2020*, Online. Association for Computational Linguistics.

Lilian Weng. 2020. How to build an open-domain question answering system?