BERT++: Reading Comprehension on SQuAD

Lirong Yuan
Stanford University
yuanzi@stanford.edu

Abstract

Question answering (QA) is a challenging task since the system need to have a wide range of skills from commonsense reasoning to coreference resolution. The paper is focused on the task of SQuAD2.0, which requires determining whether a question is answerable and extracting the correct answer from the given passage. For this paper I proposed an innovative architecture to model answerability of a question based on BERT inspired by the read and verify system described in paper from Hu et al. [2018]. The ensemble of four models achieved 75.300 EM and 77.696 F1 on test set and a competitive rank in the course leaderboard.

1 Introduction

Read Comprehension (RC) is the challenging task of finding an answer in a paragraph or a document. The system must have the skills to track lists or enumerations, comprehend mathematical operations, detect and resolve coreference, do logical reasoning and commonsense reasoning, understand analogy, spatiotemporal relations, causal relations, clause relations, and special sentence structures (Sugawara et al. [2017]). Neural approaches have gained significant popularity over the past few years within the natural language processing community as the production of large data sets allowed supervised systems to be built. Stanford Question Answering Dataset (SQuAD) is one of the first large reading comprehension datasets. It consists of questions posed by crowdworkers on 500+ Wikipedia articles. The SQuAD 1.0 Rajpurkar et al. [2016] contains 100,000+ question-answer pairs, where the answer to every question is a segment of text from the corresponding reading passage. The SQuAD 2.0 Rajpurkar et al. [2018] contains 50,000 new, unanswerable questions written adversarially by crowdworkers.

End-to-end systems now achieve promising results in the domain. The architecture proposed in Bi-Directional Attention Flow (BiDAF) (Seo et al. [2016]) introduced the central idea that the attention to flow both ways – from the context to the question and from the question to the context. Bidirectional Encoder Representations from Transformers (BERT) (Devlin et al. [2018]) pre-trained deep bidirectional representations by conditioning on bidirectional context. Fine-tuning the representations with an additional output layer, the model improved the SQuAD v1.1 question answering Test F1 to 93.2 (1.5 absolute improvement), outperforming human performance by 2.0. The best performing models in the SQuADv2.0 leader board are all extensions of BERT.

Although existing models such as BERT are performing well in SQuADv1.1, the metrics obtained for the SQuADv2.0 drops significantly in comparison, demonstrating lack of generalization and real understanding. Specifically, the system should abstain from answering when the question is unanswerable. The paper proposes to tackle this problem by improving the existing models with an answer verifier that predicts the legitimacy of the candidate answer. Compared with the baseline model, the single model BERT++ improved EM by 4.162 and F1 by 4.034 on the dev set. Section 2 gives a comprehensive overview of the models the paper is based on. Section 3 details the model architecture, key methods and algorithms. Section 4 describes the experiments conducted, discusses observations, summarizes main conclusions and describes future work.

2 Related Work

BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding (Devlin et al. [2018]) The paper introduced a new language representation model BERT that could be fine-tuned with an additional layer to create models and advanced the state-of-the-art for eleven NLP tasks. The end-to-end model for SQuAD represents the input question and paragraph as a single packed sequence and contains a linear layer with two output features: start and end positions. The training objective is the log-likelihood of the correct start and end positions. This is used as the baseline model for the paper. Details of the architecture will be described in next section.

Read + Verify: Machine Reading Comprehension with Unanswerable Questions (Hu et al. [2018]) The paper addressed the problem of unanswerable questions with a system that contains a neural reader and an answer verifier. The model of the neural reader uses attention mechanisms to encode passage and question into two representations and pointer network to predict start and end positions of candidate answers and no answer probability. Given predicted answer sentence and question, the verifier then predicts the legitimacy of the candidate answer. This is model that inspired my proposed model and will be explained in next section.

3 Approach

3.1 Baseline Model: BERT for SQuAD

The baseline model is the standard BERT model applied on SQuaAD dataset as described in the paper by Devlin et al. [2018]. The approach is to fine-tune the BERT model with an additional linear layer to predict the start and end positions of the answer. For the baseline model, I reused the existing PyTorch version of BERT made by NLP researchers from HuggingFace [1]. The script run_squad.py provided from HuggingFace is a giant file with 1085 lines, which made it very difficult to make improvements. First I refactored the code into multiple modules. Then I added checkpoint saver and loader with scores evaluated on the dev set. Next I implemented BERT++ and HybridBERT proposed below. The code is available in github [2].

3.2 BERT++

The model BERT++ is an extension of BERT. It improves the baseline model with plausible answers as input and proposes an answer verifier as inspired by the Read+Verify paper (Hu et al. [2018]). The architecture of the model is shown in Figure 1. The reader will predict plausible answer for even unanswerable questions and the verifier will check whether the candidate answer is legitimate.

Input data For input data, I modified that data processing script to include not only answerable questions, but also unanswerable cases as examples, and consider the plausible answer as golden answer to train the neural reader and answer verifier.

Neural Reader Given the input question and paragraph, first process them into a sequence of tokens (question and answer tokens) and feed them into the embedding layer of the reader to produce token embeddings with the BERT model. Next the linear layer of the reader reads the produced hidden vector and outputs two features: start and end positions using the same method as in the baseline model. The loss function is the cross entropy loss for the start and end positions. During training, the loss is averaged across the batch and the optimizer is Adam.

Answer Verifier Given the input question, paragraph and answers, the verifier will first find the sentence in paragraph that contains the answer, and the embedding layer will process it into a sequence of tokens (question and answer sentence tokens) and produce an embedding for each token with the BERT model. The classification layer of the verifier reads the pooled vector produced from BERT and outputs a sentence-level no-answer probability $P = softmax(CW^T) \in \mathbb{R}^K$, where $C \in \mathbb{R}^H$ is the final hidden state for the first token [CLS] in the input, $W \in \mathbb{R}^{K \times H}$ is a linear layer for classification,

[1] https://github.com/huggingface/pytorch-pretrained-BERT

[2] https://github.com/lirongyuan/BERTPlus/tree/master/squad

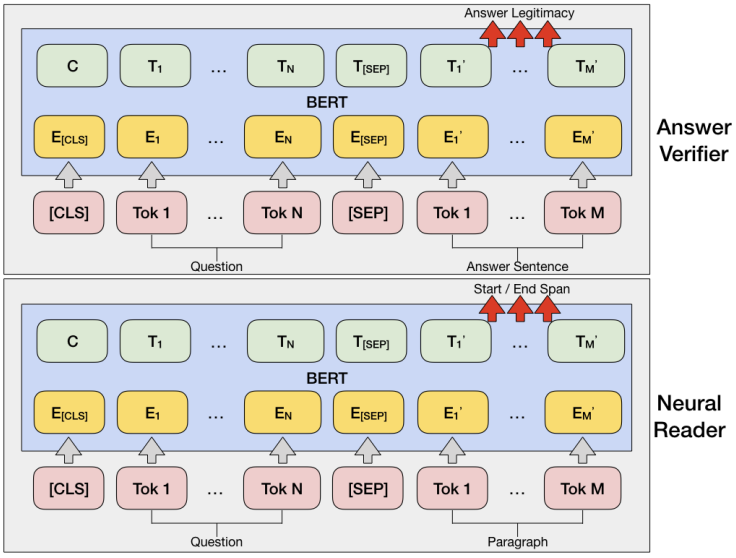

Figure 1: BERT++ model architecture

and K is the number of labels. It uses standard cross-entropy objective to minimize the negative log-likelihood of the correct labels. I implemented the data processing code to convert data into training examples, designed the verifier model to utilize BERT to predict no-answer probability, and coded script to perform training, evaluation and testing. This is a novel approach that is different from the original paper's implementation.

Predictions To predict an answer span, first processes the input into a sequence of tokens (question and answer tokens) and feeds them into the reader model to produce a candidate answer and a passage-level no-answer probability. Given the candidate answer, finds the answer sentence, processes it with question into token sequence (question and answer sentence tokens), and feeds them into the verifier model to compute a sentence-level probability. A question is marked as unanswerable if the mean of the two normalized probabilities exceeds some threshold, which is tuned to maximize F1 score on the development set. I implemented the code to glue the neural reader and answer verifier into a single pipeline to calculate final predictions.

3.3 HybridBERT

The model HybridBERT is another extension of BERT. It improves the baseline model by outputting not only start and end positions, but also a no-answer probability and updates the loss function by taking the no-answer-ability into account. It was implemented following the approach described in the paper (Clark and Gardner [2017]).

Given the input question and paragraph, first process them into a sequence of tokens (question and answer tokens) and feed them into the embedding layer to produce token embeddings with the BERT model. Next the linear layer reads the produced hidden vector and outputs two features: start and end positions; the classification layer reads the pooled hidden vector and outputs a no-answer probability. The cross entropy loss is computed as follows, where α_i and β_j are the scores for the start and end bounds produced by the model for token i and j, α_a and β_b are the augmented ground-truth answer boundaries, δ is 1 if the question is answerable and 0 otherwise and z is the no-answer probability.

$$L_{joint} = -\log\left(\frac{(1-\delta)e^{z} + \delta e^{\alpha_a \beta_b}}{e^{z} + \sum_{i=1}^{l_p}\sum_{j=1}^{l_p} e^{\alpha_i \beta_j}}\right) \tag{1}$$

4 Experiments

4.1 Data

The data set is provided in the final project github repository [3], note that the dev and test sets are slightly different from the official SQuAD 2.0.

- train (129,941 examples): All taken from the official SQuAD 2.0 training set.
- dev (6,078 examples): Roughly half of the official dev set, randomly selected.
- test (5,915 examples): The remaining examples from the official dev set, plus hand-labeled examples.

4.2 Evaluation Method

- Quantitative: performance of each model is evaluated using the two standard metrics: exact match (EM) score and F1 score. EM score measures whether the model output matches the ground truth answers exactly. F1 score is the harmonic mean of precision and recall, and less strict than then EM score. Besides EM and F1, the impact of passage length, question length, answer length on errors are also analyzed. By drawing the distribution of errors across different thresholds, it will help us better understand the model.
- Qualitative: for the best performing model, I sampled error examples, classify them based on error types, and analyze potential reasons using categories defined by Yatskar [2018].

4.3 Experiment Details

The baseline model was trained on Standard NC6 (6 vcpus, 56 GB memory) with pre-trained bert-base-cased model, batch size of 4, maximum sequence length of 384 for 2 epochs. It used Adam with learning rate of 3e-5, β_1 = 0.9, β_2 = 0.999, L2 weight decay of 0.01, learning rate warm up over the first 10,000 steps, and linear decay of the learning rate. The total running time was around 10 hours.

The BERT++ model was trained on Standard NC6sv3 (6 vcpus, 112 GB memory). The reader was trained with pre-trained bert-large-uncased model, batch size of 4, maximum sequence length of 384 for 2 epochs. It used Adam with learning rate of $3e^{-5}$, β_1 = 0.9, β_2 = 0.999, L2 weight decay of 0.01, learning rate warm up over the first 10,000 steps, and linear decay of the learning rate. The verifier was trained with a similar configuration with difference being: learning rate of $2e^{-5}$ and a dropout probability of 0.1 on the classification layer. Since two models were trained separately, the total running time was around 20 hours. For the ensemble model, I trained with varied hyper-parameters including learning rate ($2e^{-5}$, $3e^{-5}$, $5e^{-5}$), bert model (large-cased, large-uncased, base-cased, base-uncased) and epoch number (2, 3). I also experimented with training on subsets of the original training data: some readers were trained on only questions that are answerable; some verifiers were trained on a balanced data set selected from the original training data that contains roughly same number of answerable and unanswerable questions (note that in the default training set, the number of answerable questions is almost twice the number of unanwerable questions), and selected the ones that gave the best metrics on the dev set to ensemble.

The HybridBERT model was trained on Standard NC6sv3 (6 vcpus, 112 GB memory) with pre-trained bert-based-cased model, batch size of 4, maximum sequence length of 384 for 2 epochs. It used Adam with learning rate of $3e^{-5}$, β_1 = 0.9, β_2 = 0.999, L2 weight decay of 0.01, learning rate warm up over the first 10,000 steps, and linear decay of the learning rate. The total running time was around 10 hours.

4.4 Results

The quantitative results for the baseline, HybridBERT, and BERT++ models are shown in Table 1. The baseline score is lower than expected score for the BERT (single model) from Google AI Language in the standard SQuAD2.0 leaderboard (EM: 80.005, F1: 83.061). This could be due to: (1) they did a more comprehensive hyper-parameter search and reported the best score (2) the implementation details for SQuAD 2.0 are not mentioned in the paper. Among all the models I've

[3] https://github.com/chrischute/squad

Table 1: SQuAD2.0 results

System	Dev PCE Leaderboard		Test PCE Leaderboard	
	EM	F1	EM	F1
BERT (baseline)	71.915	74.487	-	-
HybridBERT (single model)	72.474	75.158	-	-
BERT++ (single large-cased model)	76.077	78.521	71.936	74.560
BERT++ (single large-uncased model)	77.657	80.180	-	-
BERT++ (ensemble model)	78.233	80.530	75.300	77.696

implemented, the best performing model is BERT++ ensemble model. This is not surprising since for HybridBERT, the prediction of answer span could interfere with the prediction of no-answer probability. When one is inaccurate, the other could be affected too. They were all submitted to the Dev and Test PCE SQuAD Leaderboard with the name "bert++" (note that this is different from another team with a similar name "BERT++"). As of March 19, 2019, BERT++ (ensemble model) achieved a competitive rank at position 8 in the Test PCE Leaderboard.

The distribution of no-answer probabilities for the baseline model is in Figure 2. For questions with no answers, the no-answer probabilities are only slightly higher than answerable questions, indicating that the baseline model could not model the answerability of a question well. The distribution of no-answer probabilities for the BERT++ model is in Figure 3. Note that the probability distribution is centered around 1.0 for unanswerable questions, and 0.0 for answerable questions, indicating that the BERT++ system could model the question answerability much better.

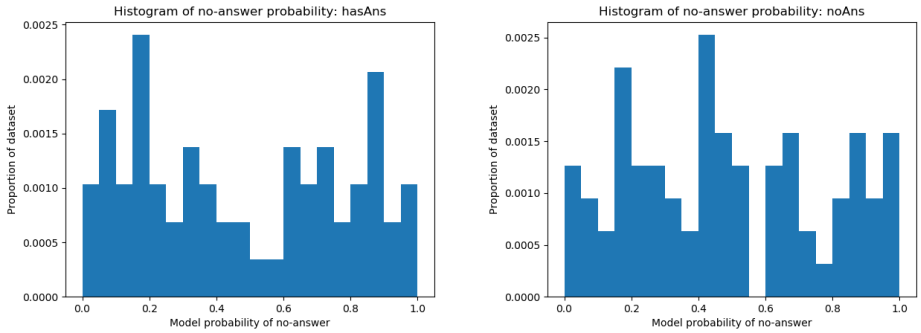

Figure 2: Histograms of no-answer probabilities for the BERT (baseline model)

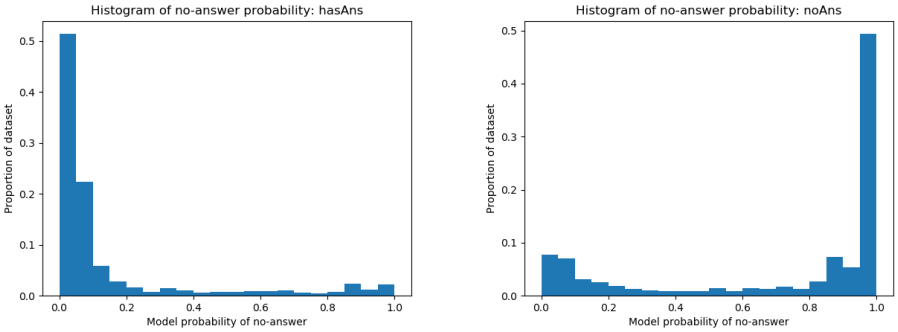

Figure 3: Histograms of no-answer probabilities for the BERT++ (single large-cased model)

Besides the span-level performance (EM and F1), I analyzed the impact of passage length, question length, answer length on errors by calculating the metrics across different thresholds:

Table 2: Impact of passage length on performance (Dev v2.0 data set)

		Baseline Model		BERT++ (single model)	
Passage length	Number of questions	EM	F1	EM	F1
[0, 640)	2099	69.271	71.705	76.560	79.410
[640, 1280)	3424	72.663	75.010	78.417	80.636
[1280, 5120)	555	76.936	81.418	77.117	80.282

Table 3: Impact of question length on performance (Dev v2.0 data set)

		Baseline Model		BERT++ (single model)	
Question length	Number of questions	EM	F1	EM	F1
[0, 40)	983	70.80	74.56	77.009	80.267
[40, 60)	2525	73.58	75.84	79.485	81.547
[60, 80)	1615	71.26	73.35	76.718	78.965
[80, 320)	955	69.52	72.53	75.078	78.532

Table 4: Impact of answer length on performance (Dev v2.0 data set)

		Baseline Model		BERT++ (single model)	
Answer length	Number of questions	EM	F1	EM	F1
[0, 5)	3467	77.09	77.20	84.741	84.828
[5, 20)	1568	69.64	73.10	71.938	75.606
[20, 160)	1043	57.90	67.35	62.703	71.605

Table 2 shows that for the baseline model, the longer the passage is, the higher the model performance; for the BERT++ model, the performance is best for passages with length between 640 and 1280. Table 3 shows that both models achieve better performance with questions that has length between 40 and 60. Table 4 shows that for both models, the shorter the answer is, the higher the model performance. Although the BERT++ model has higher scores across all dimensions, there aren't significant differences on the trends of metrics across different thresholds. This is probably because none of them was trained specifically to improve performance on subsets of the data.

4.5 Analysis

For qualitative error analysis, I sampled incorrect predictions for unanswerable questions [4] from BERT++ (large-uncased) model on dev data set and categorize them according to error types defined by Yatskar [2018]. Examples of each error type are shown in Table 5.

Here are detailed descriptions of error types and potential future improvements.

- Missing Information: the question could be answered, but the information required is not in the context (e.g. there are variables not associated with all problems solved within logarithmic space). The model fails to recognize that sentences enclosed in parentheses are used to provide a definition for the preceding word, and that L itself is the set of all problems that can be solved in logarithmic space. This could be indicative of lack of understanding for punctuation or the negation phrase "not associated". Both could be improved with more powerful language representations.
- False Premise: the question asserts a fact that contradicts information in the context (e.g. according to context, Montcalm at Quebec did not defeat anyone). The model confuses the active and passive voices of the verb: "defeated" is very different from "was defeated". To improve for the question type, we could use neural dependency parser to analyze the grammatical structure of a sentence, and encode the structure along with the token embeddings as inputs for the system.
- Topic Error: the questions references a related but different entity in the context (e.g. the question asks about "vapor turbine" while the context is about "mercury vapor turbine").

[4] Sample of incorrect predictions for **answerable questions** could be viewed in Appendix.

Table 5: Examples of error types for unanswerable questions (Dev v2.0 data set)

Error Type	Passage	Question	Predicted Answer
Missing Information	Similarly, it is not known if L (the set of all problems that can be solved in logarithmic space) is strictly contained in P or equal to P. Again, there are many complexity classes between the two, such as NL and NC, and it is not known if they are distinct or equal classes.	What variable is not associated with all problems solved within logarithmic space?	L
False premise	... James Wolfe defeated Montcalm at Quebec (in a battle that claimed the lives of both commanders), and victory at Fort Niagara successfully cut off the French frontier forts further to the west and south...	Who was defeated by Montcalm at Quebec?	James Wolfe
Topic error	... Mercury is the working fluid in the mercury vapor turbine. Low boiling hydrocarbons can be used in a binary cycle.	What is the typical working fluid in a vapor turbine?	Mercury
Content negation	... In 1018, Roger de Tosny travelled to the Iberian Peninsula to carve out a state for himself from Moorish lands, but failed...	Who carved out a state for himself from Moorish lands?	Roger de Tosny

While in some cases this is acceptable (e.g. yellow banana is the same as banana in most situations), in this case the model fails to recognize that not all vapor turbines are mercury vapor turbines. This could potentially be solvable with "enhanced representation through knowledge integration (ERNIE)[5]", which is better at learning the complete semantic representation of larger semantic units.

- Content negation: the question asks for the opposite information mentioned in the context (e.g. Roger de Tosny failed to carve out a state for himself). The model could not understand that the "but failed" negates the meaning of the whole sentence. Similar to "False Premise", a neural dependency parser should help in these cases.

4.6 Conclusion

The standard SQuAD2.0 leaderboard has demonstrated that BERT is an integral part of the best performing question answering systems. In this paper we proposed extensions of BERT: HybridBERT (inspired by Clark and Gardner [2017]) and BERT++ (inspired by Hu et al. [2018]). The major contribution is to model the answerability of a given passage and question, allowing the model to refrain from answering when the question is unanswerable. From the quantitative analysis, we learned that BERT++ is more effective than HybridBERT. The best performing model is the ensemble of four BERT++ models, which achieved 75.300 EM, 77.696 F1 on test set and a competitive rank in the class PCE test leaderboard.

While the model is better at predicting answerability of questions, future improvements are still needed to avoid errors such as missing information, false premise, topic error and content negation. Specifically, we could enhance language representation with integration of world knowledge to learn more complete semantic representation of larger language units; we could use neural dependency parser to encode sentence structures as inputs to allow better sentence representation; last but not least, we could also build specialized models that focuses on improving performances for data sets that have: (1) short passages (2) very short and very long questions (3) long answers, which would allow us to produce an ensemble model that gives weighted decisions depending on the passage length, question length and predicted answer lengths.

Acknowledgments

I would like to thank Chris Manning and all the TAs for the wonderful classes and patient help during office hours. I would also like to thank Microsoft Azure for providing resources to train the models.

[5]http://research.baidu.com/Blog/index-view?id=113

References

Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. *CoRR*, abs/1710.10723, 2017. URL http://arxiv.org/abs/1710.10723.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: pre-training of deep bidirectional transformers for language understanding. *CoRR*, abs/1810.04805, 2018. URL http://arxiv.org/abs/1810.04805.

Minghao Hu, Furu Wei, Yuxing Peng, Zhen Huang, Nan Yang, and Ming Zhou. Read + verify: Machine reading comprehension with unanswerable questions. *CoRR*, abs/1808.05759, 2018. URL http://arxiv.org/abs/1808.05759.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100, 000+ questions for machine comprehension of text. *CoRR*, abs/1606.05250, 2016. URL http://arxiv.org/abs/1606.05250.

Pranav Rajpurkar, Robin Jia, and Percy Liang. Know what you don't know: Unanswerable questions for squad. *CoRR*, abs/1806.03822, 2018. URL http://arxiv.org/abs/1806.03822.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603, 2016. URL http://arxiv.org/abs/1611.01603.

Saku Sugawara, Hikaru Yokono, and Akiko Aizawa. Prerequisite skills for reading comprehension: Multi-perspective analysis of mctest datasets and systems. In *Thirty-First AAAI Conference on Artificial Intelligence*, 2017.

Soumya Wadhwa, Khyathi Chandu, and Eric Nyberg. Comparative analysis of neural qa models on squad. In *Proceedings of the Workshop on Machine Reading for Question Answering*, pages 89–97. Association for Computational Linguistics, 2018. URL http://aclweb.org/anthology/W18-2610.

Mark Yatskar. A qualitative comparison of coqa, squad 2.0 and quac. *CoRR*, abs/1809.10735, 2018. URL http://arxiv.org/abs/1809.10735.

Appendix

For qualitative error analysis, I also sampled incorrect predictions for answerable questions from BERT++ (large-uncased) single model on dev data set and categorize them according to error types defined by Wadhwa et al. [2018]. Due to page limits, and that the model is not focused to improve performance for answerable questions, I did not include them in the main paper. Broadly, there are two categories of error types: boundary-based errors due to incorrect answer span and inference-based errors due to inability to infer the meaning of the context and question. Examples of each error type are shown in Tables 6 and 7.

Table 6: Examples of boundary-based errors for answerable questions - **bold** indicates ground truth

Error Type	Passage	Question	Predicted Answer
Incorrect answer boundary (longer)	... The Duchy of Normandy, which they formed by treaty with the French crown, was a great fief of medieval France, and under **Richard I** of Normandy was forged into a cohesive and formidable principality in feudal tenure...	Who ruled the duchy of Normandy?	Richard I of Normandy
Incorrect answer boundary (shorter)	The European Commission is the main executive body of the European Union. Article 17(1) of the Treaty on European Union states the Commission should "promote the general interest of the Union" while **Article 17(3)** adds that Commissioners should be "completely independent" and not "take instructions from any Government"...	Which article of the Treaty on European Union states that Commissioners should be completely independent and not take instructions from any Government?	Article 17(3
Soft Correct	... Warsaw hosts many events and festivals. Among the **events** worth particular attention are: the International Frédéric Chopin Piano Competition, the International Contemporary Music Festival Warsaw Autumn, the Jazz Jamboree, Warsaw Summer Jazz Days, the International Stanisław Moniuszko Vocal Competition, the Mozart Festival, and the Festival of Old Music.	Warsaw Summer Jazz Days is one of the many what hosted by Warsaw?	events and festivals

Table 7: Examples of inference-based errors for answerable questions - **bold** indicates ground truth

Error Type	Passage	Question	Predicted Answer
Multi-Sentence	An early important political response to the opening of hostilities was the convening of the Albany Congress in June and July, 1754. The goal of the congress was to formalize a unified front in trade and negotiations with various Indians, since allegiance of the various tribes and nations was seen to be pivotal in the success in the war that was unfolding. The plan that the delegates agreed to was never ratified by the colonial legislatures nor approved of by the crown. Nevertheless, the format of the congress and many specifics of the plan **became the prototype for confederation during the War of Independence**.	What was the importance of the congress?	to formalize a unified front in trade and negotiations with various Indians
Same Entity Type Confusion	... In many species, the immune system can be classified into subsystems, such as the innate immune system versus the adaptive immune system, or **humoral immunity versus cell-mediated immunity**...	What are the two different types of immunity?	innate immune system versus the adaptive immune system, or humoral immunity versus cell-mediated immunity
Missing Inference	The Presiding Officer (or Deputy Presiding Officer) decides who speaks in chamber debates and **the amount of time for which they are allowed to speak**...	What is also decided by the Presiding Officer?	who speaks in chamber debates