Which Linguist Invented the Lightbulb? Presupposition Verification for Question-Answering

Najoung Kim†,*, **Ellie Pavlick**$^{\phi,\delta}$, **Burcu Karagol Ayan**$^{\delta}$, **Deepak Ramachandran**$^{\delta,*}$
†Johns Hopkins University $^{\phi}$Brown University $^{\delta}$Google Research
n.kim@jhu.edu {epavlick,burcuka,ramachandrand}@google.com

Abstract

Many Question-Answering (QA) datasets contain unanswerable questions, but their treatment in QA systems remains primitive. Our analysis of the Natural Questions (Kwiatkowski et al., 2019) dataset reveals that a substantial portion of unanswerable questions (∼21%) can be explained based on the presence of *unverifiable presuppositions*. Through a user preference study, we demonstrate that the oracle behavior of our proposed system—which provides responses based on presupposition failure—is preferred over the oracle behavior of existing QA systems. Then, we present a novel framework for implementing such a system in three steps: presupposition generation, presupposition verification, and explanation generation, reporting progress on each. Finally, we show that a simple modification of adding presuppositions and their verifiability to the input of a competitive end-to-end QA system yields modest gains in QA performance and unanswerability detection, demonstrating the promise of our approach.

1 Introduction

Many Question-Answering (QA) datasets including Natural Questions (NQ) (Kwiatkowski et al., 2019) and SQuAD 2.0 (Rajpurkar et al., 2018) contain questions that are *unanswerable*. While unanswerable questions constitute a large part of existing QA datasets (e.g., 51% of NQ, 36% of SQuAD 2.0), their treatment remains primitive. That is, (closed-book) QA systems label these questions as *Unanswerable* without detailing why, as in (1):

(1) a. **Answerable Q:** Who is the current monarch of the UK?
System: Elizabeth II.

b. **Unanswerable Q:** Who is the current monarch of France?
System: Unanswerable.

Unanswerability in QA arises due to a multitude of reasons including retrieval failure and malformed questions (Kwiatkowski et al., 2019). We focus on a subset of unanswerable questions—namely, questions containing failed *presuppositions* (background assumptions that need to be satisfied).

Questions containing failed presuppositions do not receive satisfactory treatment in current QA. Under a setup that allows for *Unanswerable* as an answer (as in several closed-book QA systems; Figure 1, left), the best case scenario is that the system correctly identifies that a question is unanswerable and gives a generic, unsatisfactory response as in (1-b). Under a setup that does not allow for *Unanswerable* (e.g., open-domain QA), a system's attempt to answer these questions results in an inaccurate accommodation of false presuppositions. For example, Google answers the question *Which linguist invented the lightbulb?* with *Thomas Edison*, and Bing answers the question *When did Marie Curie discover Uranium?* with *1896* (retrieved Jan 2021). These answers are clearly inappropriate, because answering these questions with *any* name or year endorses the false presuppositions *Some linguist invented the lightbulb* and *Marie Curie discovered Uranium*. Failures of this kind are extremely noticeable and have recently been highlighted by social media (Munroe, 2020), showing an outsized importance regardless of their effect on benchmark metrics.

We propose a system that takes presuppositions into consideration through the following steps (Figure 1, right):

1. **Presupposition generation:** *Which linguist invented the lightbulb?* → *Some linguist invented the lightbulb.*

[*] Corresponding authors, [†] Work done at Google

Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, pages 3932–3945
August 1–6, 2021. ©2021 Association for Computational Linguistics

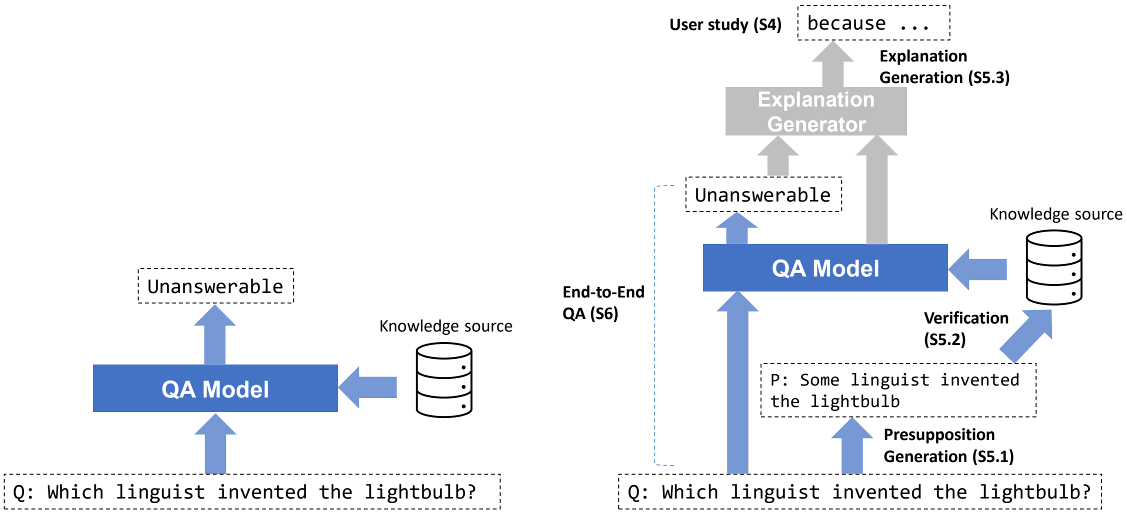

Unanswerable
QA Model
Knowledge source
Q: Which linguist invented the lightbulb?
User study (S4)
because ...
Explanation Generation (S5.3)
Explanation Generator
Unanswerable
End-to-End QA (S6)
QA Model
Knowledge source
Verification (S5.2)
P: Some linguist invented the lightbulb
Presupposition Generation (S5.1)
Q: Which linguist invented the lightbulb?

Figure 1: A comparison of existing closed-book QA pipelines (left) and the proposed QA pipeline in this work (right). The gray part of the pipeline is only manually applied in this work to conduct headroom analysis.

2. **Presupposition verification:** *Some linguist invented the lightbulb.* → Not verifiable
3. **Explanation generation:** (*Some linguist invented the lightbulb*, Not verifiable) → *This question is unanswerable because there is insufficient evidence that any linguist invented the lightbulb.*

Our contribution can be summarized as follows:

- We identify a subset of unanswerable questions—questions with failed presuppositions—that are not handled well by existing QA systems, and quantify their role in naturally occurring questions through an analysis of the NQ dataset (S2, S3).
- We outline how a better QA system could handle questions with failed presuppositions, and validate that the oracle behavior of this proposed system is more satisfactory to users than the oracle behavior of existing systems through a user preference study (S4).
- We propose a novel framework for handling presuppositions in QA, breaking down the problem into three parts (see steps above), and evaluate progress on each (S5). We then integrate these steps end-to-end into a competitive QA model and achieve modest gains (S6).

2 Presuppositions

Presuppositions are implicit assumptions of utterances that interlocutors take for granted. For example, if I uttered the sentence *I love my hedgehog*, it is assumed that I, the speaker, do in fact own a hedgehog. If I do not own one (hence the presupposition fails), uttering this sentence would be inappropriate. Questions may also be inappropriate in the same way when they contain failed presuppositions, as in the question *Which linguist invented the lightbulb?*.

Presuppositions are often associated with specific words or syntactic constructions ('triggers'). We compiled an initial list of presupposition triggers based on Levinson (1983: 181–184) and Van der Sandt (1992),[1] and selected the following triggers based on their frequency in NQ (» means 'presupposes'):

- Question words (*what, where, who...*): *Who did Jane talk to? » Jane talked to someone.*
- Definite article (*the*): *I saw the cat » There exists some contextually salient, unique cat.*
- Factive verbs (*discover, find out, prove...*): *I found out that Emma lied. » Emma lied.*
- Possessive *'s*: *She likes Fred's sister. » Fred has a sister.*
- Temporal adjuncts (*when, during, while...*): *I was walking when the murderer escaped from prison. » The murderer escaped from prison.*
- Counterfactuals (*if* + past): *I would have been happier if I had a dog. » I don't have a dog.*

Our work focuses on presuppositions of questions. We assume presuppositions *project* from

[1] We note that it is a simplifying view to treat all triggers under the banner of presupposition; see Karttunen (2016).

Cause of unanswerability	%	Example Q	Comment
Unverifiable presupposition	30%	*what is the stock symbol for mars candy*	Presupposition *'stock symbol for mars candy exists'* fails
Reference resolution failure	9%	*what kind of vw jetta do i have*	The system does not know who 'i' is
Retrieval failure	6%	*when did the salvation army come to australia*	Page retrieved was *Safe Schools Coalition Australia*
Subjectivity	3%	*what is the perfect height for a model*	Requires subjective judgment
Commonsensical	3%	*where does how to make an american quilt take place*	Document contains no evidence that the movie took place somewhere, but it is commonsensical that it did
Actually answerable	8%	*when do other cultures celebrate the new year*	The question was actually answerable given the document
Not a question/Malformed question	3%	*where do you go my lovely full version*	Not an actual question

Table 1: Example causes of unanswerability in NQ. % denotes the percentage of questions that both annotators agreed to be in the respective cause categories.

wh-questions—that is, presuppositions (other than the presupposition introduced by the interrogative form) remain constant under *wh*-questions as they do under negation (e.g., *I don't like my sister* has the same possessive presupposition as *I like my sister*). However, the projection problem is complex; for instance, when embedded under other operators, presuppositions can be overtly denied (Levinson 1983: 194). See also Schlenker (2008), Abrusán (2011), Schwarz and Simonenko (2018), Theiler (2020), *i.a.,* for discussions regarding projection patterns under *wh*-questions. We adopt the view of Strawson (1950) that definite descriptions presuppose both existence and (contextual) uniqueness, but this view is under debate. See Coppock and Beaver (2012), for instance, for an analysis of *the* that does not presuppose existence and presupposes a weaker version of uniqueness. Furthermore, we currently do not distinguish predicative and argumental definites.

Presuppositions and unanswerability. Questions containing failed presuppositions are often treated as unanswerable in QA datasets. An example is the question *What is the stock symbol for Mars candy?* from NQ. This question is not answerable with any description of a stock symbol (that is, an answer to the *what* question), because Mars is not a publicly traded company and thus does not have a stock symbol. A better response would be to point out the presupposition failure, as in *There is no stock symbol for Mars candy*. However, statements about negative factuality are rarely explicitly stated, possibly due to reporting bias (Gordon and Van Durme, 2013). Therefore, under an extractive QA setup as in NQ where the answers are spans from an answer source (e.g., a Wikipedia article), it is likely that such questions will be unanswerable.

Our proposal is based on the observation that the denial of a failed presupposition (¬P) can be used to explain the unanswerability of questions (Q) containing failed presuppositions (P), as in (2).

(2) **Q:** Who is the current monarch of France?
P: There is a current monarch of France.
¬P: There is no such thing as a current monarch of France.

An answer that refers to the presupposition, such as ¬P, would be more informative compared to both *Unanswerable* (1-b) and an extractive answer from documents that are topically relevant but do not mention the false presupposition.

3 Analysis of Unanswerable Questions

First, to quantify the role of presupposition failure in QA, two of the authors analyzed 100 randomly selected unanswerable wh-questions in the NQ development set.[2] The annotators labeled each question as *presupposition failure* or *not presupposition failure*, depending on whether its unanswerability could be explained by the presence of an unverifiable presupposition with respect to the associated document. If the unanswerability could not be explained in terms of presupposition failure, the annotators provided a reasoning. The Cohen's κ for inter-annotator agreement was 0.586.

We found that 30% of the analyzed questions could be explained by the presence of an unverifiable presupposition in the question, considering only the cases where both annotators were in agreement (see Table 1).[3] After adjudicating the reasoning about unanswerability for the non-presupposition failure cases, another 21% fell into cases where presupposition failure could be partially informative (see Table 1 and Appendix A for details). The unverifiable presuppositions were

[2] The NQ development set provides 5 answer annotations per question—we only looked at questions with 5/5 Null answers here.

[3] wh-questions constitute ∼69% of the NQ development set, so we expect the actual portion of questions with presupposition failiure-based explanation to be ∼21%.

Question: *where can i buy a japanese dwarf flying squirrel*	
Simple unanswerable	*This question is unanswerable.*
Presupposition failure-based	*This question is unanswerable because we could not verify that you can buy a Japanese Dwarf Flying Squirrel anywhere.*
Extractive explanation	*This question is unanswerable because it grows to a length of 20 cm (8 in) and has a membrane connecting its wrists and ankles which enables it to glide from tree to tree.*
DPR rewrite	*After it was returned for the second time, the original owner, referring to it as "the prodigal gnome", said she had decided to keep it and would not sell it on Ebay again.*

Table 2: Systems (answer types) compared in the user preference study and examples.

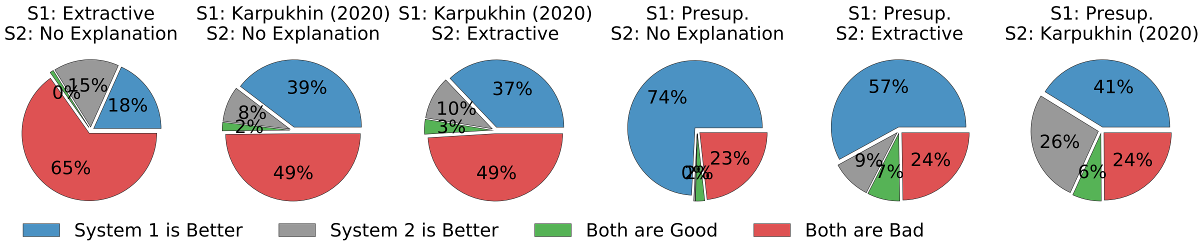


Figure 2: Results of the user preference study. Chart labels denote the two systems being compared (S1 vs. S2).

triggered by question words (19/30), the definite article *the* (10/30), and a factive verb (1/30).

4 User Study with Oracle Explanation

Our hypothesis is that statements explicitly referring to failed presuppositions can better[4] speak to the unanswerability of corresponding questions. To test our hypothesis, we conducted a side-by-side comparison of the oracle output of our proposed system and the oracle output of existing (closed-book) QA systems for unanswerable questions. We included two additional systems for comparison; the four system outputs compared are described below (see Table 2 for examples):

- **Simple unanswerable:** A simple assertion that the question is unanswerable (i.e., *This question is unanswerable*). This is the oracle behavior of closed-book QA systems that allow *Unanswerable* as an answer.
- **Presupposition failure-based explanation:** A denial of the presupposition that is unverifiable from the answer source. This takes the form of either *This question is unanswerable because we could not verify that...* or *...because it is unclear that...* depending on the type of the failed presupposition. See Section 5.3 for more details.
- **Extractive explanation:** A random sentence from a Wikipedia article that is topically related to the question, prefixed by *This question is unanswerable because...*. This system is introduced as a control to ensure that length bias is not in play in the main comparison (e.g., users may *a priori* prefer longer, topically-related answers over short answers). That is, since our system, **Presupposition failure-based explanation**, yields strictly longer answers than **Simple unanswerable**, we want to ensure that our system is not preferred merely due to length rather than answer quality.
- **Open-domain rewrite:** A rewrite of the non-oracle output taken from the demo[5] of Dense Passage Retrieval (DPR; Karpukhin et al. 2020), a competitive open-domain QA system. This system is introduced to test whether presupposition failure can be easily addressed by expanding the answer source, since a single Wikipedia article was used to determine presupposition failure. If presupposition failure is a problem particular only to closed-book systems, a competitive open-domain system would suffice to address this issue. While the outputs compared are not oracle, this system

[4] We define *better* as user preference in this study, but other dimensions could also be considered such as trustworthiness.

[5] http://qa.cs.washington.edu:2020/

Question (input)	Template	Presupposition (output)
which philosopher advocated the idea of return to nature	some __	some *philosopher advocated the idea of return to nature*
when was it discovered that the sun rotates	__	*the sun rotates*
when is the year of the cat in chinese zodiac	__ exists	*'year of the cat in chinese zodiac'* exists
when is the year of the cat in chinese zodiac	__ is contextually unique	*'year of the cat in chinese zodiac'* is contextually unique
what do the colors on ecuador's flag mean	__ has __	*'ecuador'* has *'flag'*

Table 3: Example input-output pairs of our presupposition generator. Text in italics denotes the part taken from the original question, and the plain text is the part from the generation template. All questions are taken from NQ.

has an advantage of being able to refer to all of Wikipedia. The raw output was rewritten to be well-formed, so that it was not unfairly disadvantaged (see Appendix B.2).

Study. We conducted a side-by-side study with 100 unanswerable questions. These questions were unanswerable questions due to presupposition failure, as judged independently and with high confidence by two authors.[6] We presented an exhaustive binary comparison of four different types of answers for each question (six binary comparisons per question). We recruited five participants on an internal crowdsourcing platform at Google, who were presented with all binary comparisons for all questions. All comparisons were presented in random order, and the sides that the comparisons appeared in were chosen at random. For each comparison, the raters were provided with an unanswerable question, and were asked to choose the system that yielded the answer they preferred (either *System 1* or *2*). They were also given the options *Both answers are good/bad*. See Appendix B.1 for additional details about the task setup.

Results. Figure 2 shows the user preferences for the six binary comparisons, where blue and gray denote preferences for the two systems compared. We find that presupposition-based answers are preferred against all three answer types with which they were compared, and prominently so when compared to the oracle behavior of existing closed-book QA systems (4th chart, Presup. vs. No Explanation). This supports our hypothesis that presupposition failure-based answers would be more satisfactory to the users, and suggests that building a QA system that approaches the oracle behavior of our proposed system is a worthwhile pursuit.

5 Model Components

Given that presupposition failure accounts for a substantial proportion of unanswerable questions (Section 3) and our proposed form of explanations is useful (Section 4), how can we build a QA system that offers such explanations? We decompose this task into three smaller sub-tasks: presupposition generation, presupposition verification, and explanation generation. Then, we present progress towards each subproblem using NQ.[7] We use a templatic approach for the first and last steps. The second step involves verification of the generated presuppositions of the question against an answer source, for which we test four different strategies: zero-shot transfer from Natural Language Inference (NLI), an NLI model finetuned on verification, zero-shot transfer from fact verification, and a rule-based/NLI hybrid model. Since we used NQ, our models assume a closed-book setup with a single document as the source of verification.

5.1 Step 1: Presupposition Generation

Linguistic triggers. Using the linguistic triggers discussed in Section 2, we implemented a rule-based generator to templatically generate presuppositions from questions. See Table 3 for examples, and Appendix C for a full list.

Generation. The generator takes as input a constituency parse tree of a question string from the Berkeley Parser (Petrov et al., 2006) and applies trigger-specific transformations to generate the presupposition string (e.g., taking the sentential complement of a factive verb). If there are multiple triggers in a single question, all presuppositions corresponding to the triggers are generated. Thus, a single question may have multiple presuppositions. See Table 3 for examples of input questions and output presuppositions.

[6] Hence, this set did not necessarily overlap with the randomly selected unanswerable questions from Section 3; we wanted to specifically find a set of questions that were representative of the phenomena we address in this work.

[7] Code and data will be available at https://github.com/google-research/google-research/presup-qa

How good is our generation? We analyzed 53 questions and 162 generated presuppositions to estimate the quality of our generated presuppositions. This set of questions contained at least 10 instances of presuppositions pertaining to each category. One of the authors manually validated the generated presuppositions. According to this analysis, 82.7% (134/162) presuppositions were valid presuppositions of the question. The remaining cases fell into two broad categories of error: ungrammatical (11%, 18/162) or grammatical but not presupposed by the question (6.2%, 10/162). The latter category of errors is a limitation of our rule-based generator that does not take semantics into account, and suggests an avenue by which future work can yield improvements. For instance, we uniformly apply the template *'A'* has *'B'*[8] for presuppositions triggered by *'s*. While this template works well for cases such as *Elsa's sister* » *'Elsa' has 'sister'*, it generates invalid presuppositions such as *Bachelor's degree* » #*'Bachelor' has 'degree'*. Finally, the projection problem is another limitation. For example, *who does pip believe is estella's mother* has an embedded possessive under a nonfactive verb *believe*, but our generator would nevertheless generate *'estella' has 'mother'*.

5.2 Step 2: Presupposition Verification

The next step is to verify whether presuppositions of a given question is verifiable from the answer source. The presuppositions were first generated using the generator described in Section 5.1, and then manually repaired to create a verification dataset with gold presuppositions. This was to ensure that verification performance is estimated without a propagation of error from the previous step. Generator outputs that were not presupposed by the questions were excluded.

To obtain the verification labels, two of the authors annotated 462 presuppositions on their binary verifiability (*verifiable/not verifiable*) based on the Wikipedia page linked to each question (the links were provided in NQ). A presupposition was labeled *verifiable* if the page contained any statement that either asserted or implied the content of the presupposition. The Cohen's κ for inter-annotator agreement was 0.658. The annotators reconciled the disagreements based on a post-annotation discussion to finalize the labels to be used in the experiments. We divided the annotated presuppositions into development ($n = 234$) and test ($n = 228$) sets.[9] We describe below four different strategies we tested.

Zero-shot NLI. NLI is a classification task in which a model is given a premise-hypothesis pair and asked to infer whether the hypothesis is entailed by the premise. We formulate presupposition verification as NLI by treating the document as the premise and the presupposition to verify as the hypothesis. Since Wikipedia articles are often larger than the maximum premise length that NLI models can handle, we split the article into sentences and created n premise-hypothesis pairs for an article with n sentences. Then, we aggregated these predictions and labeled the hypothesis (the presupposition) as verifiable if there are at least k sentences from the document that supported the presupposition. If we had a perfect verifier, $k = 1$ would suffice to perform verification. We used $k = 1$ for our experiments, but k could be treated as a hyperparameter. We used ALBERT-xxlarge (Lan et al., 2020) finetuned on MNLI (Williams et al., 2018) and QNLI (Wang et al., 2019) as our NLI model.

Finer-tuned NLI. Existing NLI datasets such as QNLI contain a broad distribution of entailment pairs. We adapted the model further to the distribution of entailment pairs that are specific to our generated presuppositions (e.g., *Hypothesis: NP is contextually unique*) through additional finetuning (i.e., *finer-tuning*). Through crowdsourcing on an internal platform, we collected entailment labels for 15,929 (presupposition, sentence) pairs, generated from 1000 questions in NQ and 5 sentences sampled randomly from the corresponding Wikipedia pages. We continued training the model fine-tuned on QNLI on this additional dataset to yield a finer-tuned NLI model. Finally, we aggregated per-sentence labels as before to get verifiability labels for (presupposition, document) pairs.

Zero-shot FEVER. FEVER is a fact verification task proposed by Thorne et al. (2018). We formulate presupposition verification as a fact verification task by treating the Wikipedia article as the evidence source and the presupposition as the claim. While typical FEVER systems have a docu-

[8] We used a template that puts possessor and possessee NPs in quotes instead of using different templates depending on posessor/possessee plurality (e.g., *A __ has a __/A __ has __/__ have a __/__ have __*).

[9] The dev/test set sizes did not exactly match because we kept presuppositions of same question within the same split, and each question had varying numbers of presuppositions.

Model	Macro F1	Acc.
Majority class	0.44	0.78
Zero-shot NLI (ALBERT MNLI + Wiki sentences)	0.50	0.51
Zero-shot NLI (ALBERT QNLI + Wiki sentences)	0.55	0.73
Zero-shot FEVER (KGAT + Wiki sentences)	0.54	0.66
Finer-tuned NLI (ALBERT QNLI + Wiki sentences)	0.58	0.76
Rule-based/NLI hybrid (ALBERT QNLI + Wiki presuppositions)	0.58	0.71
Rule-based/NLI hybrid (ALBERT QNLI + Wiki sentences + Wiki presuppositions)	0.59	0.77
Finer-tuned, rule-based/NLI hybrid (ALBERT QNLI + Wiki sentences + Wiki presuppositions)	**0.60**	**0.79**

Table 4: Performance of verification models tested. Models marked with 'Wiki sentence' use sentences from Wikipedia articles as premises, and 'Wiki presuppositions', generated presuppositions from Wikipedia sentences.

ment retrieval component, we bypass this step and directly perform evidence retrieval on the article linked to the question. We used the Graph Neural Network-based model of Liu et al. (2020) (KGAT) that achieves competitive performance on FEVER. A key difference between KGAT and NLI models is that KGAT can consider pieces of evidence jointly, whereas with NLI, the pieces of evidence are verified independently and aggregated at the end. For presuppositions that require multihop reasoning, KGAT may succeed in cases where aggregated NLI fails—e.g., for uniqueness. That is, if there is no sentence in the document that bears the same uniqueness presupposition, one would need to reason over all sentences in the document.

Rule-based/NLI hybrid. We consider a rule-based approach where we apply the same generation method described in Section 5 to the Wikipedia documents to extract the presuppositions of the evidence sentences. The intended effect is to extract content that is directly relevant to the task at hand—that is, we are making the presuppositions of the documents explicit so that they can be more easily compared to presuppositions being verified. However, a naïve string match between presuppositions of the document and the questions would not work, due to stylistic differences (e.g., definite descriptions in Wikipedia pages tend to have more modifiers). Hence, we adopted a hybrid approach where the zero-shot QNLI model was used to verify (document presupposition, question presupposition) pairs.

Results. Our results (Table 4) suggest that presupposition verification is challenging to existing models, partly due to class imbalance. Only the model that combines finer-tuning and rule-based document presuppositions make modest improvement over the majority class baseline (78% → 79%). Nevertheless, gains in F1 were substantial for all models (44% → 60% in best model), showing that these strategies do impact verifiability, albeit with headroom for improvement. QNLI provided the most effective zero-shot transfer, possibly because of domain match between our task and the QNLI dataset—they are both based on Wikipedia. The FEVER model was unable to take advantage of multihop reasoning to improve over (Q)NLI, whereas using document presuppositions (Rule-based/NLI hybrid) led to gains over NLI alone.

5.3 Step 3: Explanation Generation

We used a template-based approach to explanation generation: we prepended the templates *This question is unanswerable because we could not verify that...* or *...because it is unclear that...* to the unverifiable presupposition (3). Note that we worded the template in terms of *unverifiability* of the presupposition, rather than asserting that it is false. Under a closed-book setup like NQ, the only ground truth available to the model is a single document, which leaves a possibility that the presupposition is verifiable outside of the document (except in the rare occasion that it is refuted by the document). Therefore, we believe that unverifiability, rather than failure, is a phrasing that reduces false negatives.

(3) **Q:** *when does back to the future part 4 come out*
Unverifiable presupposition: there is some point in time that *back to the future part 4 comes out*
Simple prefixing: This question is unanswerable because we could not verify that *there is some point in time that back to the future part 4 comes out.*

Model	Average F1	Long answer F1	Short answer F1	Unans. Acc	Unans. F1
ETC (our replication)	0.645	0.742	0.548	0.695	0.694
+ Presuppositions (flat)	0.641	0.735	0.547	0.702	0.700
+ Verification labels (flat)	0.645	0.742	0.547	0.687	0.684
+ Presups + labels (flat)	0.643	**0.744**	0.544	0.702	0.700
+ Presups + labels (structured)	**0.649**	0.743	**0.555**	**0.703**	**0.700**

Table 5: Performance on NQ development set with ETC and ETC augmented with presupposition information. We compare our augmentation results against our own replication of Ainslie et al. (2020) (first row).

For the user study (Section 4), we used a manual, more fluent rewrite of the explanation generated by simple prefixing. In future work, fluency is a dimension that can be improved over templatic generation. For example, for (3), a fluent model could generate the response: *This question is unanswerable because we could not verify that Back to the Future Part 4 will ever come out.*

6 End-to-end QA Integration

While the 3-step pipeline is designed to generate explanations for unanswerability, the generated presuppositions and their verifiability can also provide useful guidance even for a standard extractive QA system. They may prove useful both to unanswerable and answerable questions, for instance by indicating which tokens of a document a model should attend to. We test several approaches to augmenting the input of a competitive extractive QA system with presuppositions and verification labels.

Model and augmentation. We used Extended Transformer Construction (ETC) (Ainslie et al., 2020), a model that achieves competitive performance on NQ, as our base model. We adopted the configuration that yielded the best reported NQ performance among ETC-base models.[10] We experiment with two approaches to encoding the presupposition information. First, in the *flat model*, we simply augment the input question representation (token IDs of the question) by concatenating the token IDs of the generated presuppositions and the verification labels (0 or 1) from the ALBERT QNLI model. Second, in the *structured model* (Figure 4), we take advantage of the global input layer of ETC that is used to encode the discourse units of large documents like paragraphs. Global tokens *attend* (via self-attention) to all tokens of their internal text, but for other text in the document, they only attend to the corresponding global tokens. We add one global token for each presupposition, and allow the presupposition tokens to only attend to each other and the global token. The value of the global token is set to the verification label (0 or 1).

Metrics. We evaluated our models on two sets of metrics: NQ performance (Long Answer, Short Answer, and Average F1) and Unanswerability Classification (Accuracy and F1).[11] We included the latter because our initial hypothesis was that sensitivity to presuppositions of questions would lead to better handling of unanswerable questions. The ETC NQ model has a built-in answer type classification step which is a 5-way classification between {*Unanswerable, Long Answer, Short Answer, Yes, No*}. We mapped the classifier outputs to binary answerability labels by treating the predicted label as *Unanswerable* only if its logit was greater than the sum of all other options.

Results and Discussion Table 5 shows that augmentations that use only the presuppositions or only the verification labels do not lead to gains in NQ performance over the baseline, but the presuppositions do lead to gains on Unanswerability Classification. When both presuppositions and their verifiability are provided, we see minor gains in Average F1 and Unanswerability Classification.[12] For Unanswerability Classification, the improved accuracy is different from the baseline at the 86% (flat) and 89% (structured) confidence level using McNemar's test. The main bottleneck of our model is the quality of the verification labels used for augmentation (Table 4)—noisy labels limit the capacity of the QA model to attend to the augmentations.

While the gain on Unanswerability Classification is modest, an error analysis suggests that

[10] The reported results in Ainslie et al. (2020) are obtained using a custom modification to the inference procedure that we do not incorporate into our pipeline, since we are only interested in the relative gains from presupposition verification.

[11] Here, we treated $\geq$ 4 Null answers as unanswerable, following the definition in Kwiatkowski et al. (2019).

[12] To contextualize our results, a recently published NQ model (Ainslie et al., 2020) achieved a gain of around ~2%.

the added presuppositions modulate the prediction change in our best-performing model (structured) from the baseline ETC model. Looking at the cases where changes in model prediction (i.e., *Unanswerable (U)* ↔ *Answerable (A)*) lead to correct answers, we observe an asymmetry in the two possible directions of change. The number of correct *A* → *U* cases account for 11.9% of the total number of unanswerable questions, whereas correct *U* → *A* cases account for 6.7% of answerable questions. This asymmetry aligns with the expectation that the presupposition-augmented model should achieve gains through cases where unverified presuppositions render the question unanswerable. For example, given the question *who played david brent's girlfriend in the office* that contains a false presupposition *David Brent has a girlfriend*, the structured model changed its prediction to *Unanswerable* from the base model's incorrect answer *Julia Davis* (an actress, not David Brent's girlfriend according to the document: ... *arrange a meeting with the second woman (voiced by Julia Davis)*). On the other hand, such an asymmetry is not observed in cases where changes in model prediction results in incorrect answers: incorrect *A* → *U* and *U* → *A* account for 9.1% and 9.2%, respectively. More examples are shown in Appendix F.

7 Related Work

While presuppositions are an active topic of research in theoretical and experimental linguistics (Beaver, 1997; Simons, 2013; Schwarz, 2016, *i.a.,*), comparatively less attention has been given to presuppositions in NLP (but see Clausen and Manning (2009) and Tremper and Frank (2011)). More recently, Cianflone et al. (2018) discuss automatically detecting presuppositions, focusing on adverbial triggers (e.g., *too, also...*), which we excluded due to their infrequency in NQ. Jeretic et al. (2020) investigate whether inferences triggered by presuppositions and implicatures are captured well by NLI models, finding mixed results.

Regarding unanswerable questions, their importance in QA (and therefore their inclusion in benchmarks) has been argued by works such as Clark and Gardner (2018) and Zhu et al. (2019). The analysis portion of our work is similar in motivation to unanswerability analyses in Yatskar (2019) and Asai and Choi (2020)—to better understand the causes of unanswerability in QA. Hu et al. (2019); Zhang et al. (2020); Back et al. (2020) consider answerability detection as a core motivation of their modeling approaches and propose components such as independent no-answer losses, answer verification, and answerability scores for answer spans.

Our work is most similar to Geva et al. (2021) in proposing to consider implicit assumptions of questions. Furthermore, our work is complementary to QA explanation efforts like Lamm et al. (2020) that only consider answerable questions.

Finally, abstractive QA systems (e.g., Fan et al. 2019) were not considered in this work, but their application to presupposition-based explanation generation could be an avenue for future work.

8 Conclusion

Through an NQ dataset analysis and a user preference study, we demonstrated that a significant portion of unanswerable questions can be answered more effectively by calling out unverifiable presuppositions. To build models that provide such an answer, we proposed a novel framework that decomposes the task into subtasks that can be connected to existing problems in NLP: presupposition identification (parsing and text generation), presupposition verification (textual inference and fact verification), and explanation generation (text generation). We observed that presupposition verification, especially, is a challenging problem. A combination of a competitive NLI model, finer-tuning and rule-based hybrid inference gave substantial gains over the baseline, but was still short of a fully satisfactory solution. As a by-product, we showed that verified presuppositions can modestly improve the performance of an end-to-end QA model.

In the future, we plan to build on this work by proposing QA systems that are more robust and cooperative. For instance, different types of presupposition failures could be addressed by more fluid answer strategies—e.g., violation of uniqueness presuppositions may be better handled by providing all possible answers, rather than stating that the uniqueness presupposition was violated.

Acknowledgments

We thank Tom Kwiatkowski, Mike Collins, Tania Rojas-Esponda, Eunsol Choi, Annie Louis, Michael Tseng, Kyle Rawlins, Tania Bedrax-Weiss, and Elahe Rahimtoroghi for helpful discussions about this project. We also thank Lora Aroyo for help with user study design, and Manzil Zaheer for pointers about replicating the ETC experiments.

References

Márta Abrusán. 2011. Presuppositional and negative islands: a semantic account. *Natural Language Semantics*, 19(3):257–321.

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. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 268–284, Online. Association for Computational Linguistics.

Akari Asai and Eunsol Choi. 2020. Challenges in information seeking QA: Unanswerable questions and paragraph retrieval. *arXiv:2010.11915*.

Seohyun Back, Sai Chetan Chinthakindi, Akhil Kedia, Haejun Lee, and Jaegul Choo. 2020. NeurQuRI: Neural question requirement inspector for answerability prediction in machine reading comprehension. In *International Conference on Learning Representations*.

David Beaver. 1997. Presupposition. In *Handbook of Logic and Language*, pages 939–1008. Elsevier.

Andre Cianflone, Yulan Feng, Jad Kabbara, and Jackie Chi Kit Cheung. 2018. Let's do it "again": A first computational approach to detecting adverbial presupposition triggers. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 2747–2755, Melbourne, Australia. Association for Computational Linguistics.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 845–855, Melbourne, Australia. Association for Computational Linguistics.

David Clausen and Christopher D. Manning. 2009. Presupposed content and entailments in natural language inference. In *Proceedings of the 2009 Workshop on Applied Textual Inference (TextInfer)*, pages 70–73, Suntec, Singapore. Association for Computational Linguistics.

Elizabeth Coppock and David Beaver. 2012. Weak uniqueness: The only difference between definites and indefinites. In *Semantics and Linguistic Theory*, volume 22, pages 527–544.

Angela Fan, Yacine Jernite, Ethan Perez, David Grangier, Jason Weston, and Michael Auli. 2019. ELI5: Long form question answering. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3558–3567, Florence, Italy. Association for Computational Linguistics.

Mor Geva, Daniel Khashabi, Elad Segal, Tushar Khot, Dan Roth, and Jonathan Berant. 2021. Did Aristotle use a laptop? A question answering benchmark with implicit reasoning strategies. *arXiv:2101.02235*.

Jonathan Gordon and Benjamin Van Durme. 2013. Reporting bias and knowledge acquisition. In *Proceedings of the 2013 Workshop on Automated Knowledge Base Construction*, AKBC '13, page 25–30, New York, NY, USA. Association for Computing Machinery.

Minghao Hu, Furu Wei, Yuxing Peng, Zhen Huang, Nan Yang, and Dongsheng Li. 2019. Read+ verify: Machine reading comprehension with unanswerable questions. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 6529–6537.

Paloma Jeretic, Alex Warstadt, Suvrat Bhooshan, and Adina Williams. 2020. Are natural language inference models IMPPRESsive? Learning IMPlicature and PRESupposition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 8690–8705, Online. Association for Computational Linguistics.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 6769–6781, Online. Association for Computational Linguistics.

Lauri Karttunen. 2016. Presupposition: What went wrong? In *Semantics and Linguistic Theory*, volume 26, pages 705–731.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: A benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:452–466.

Matthew Lamm, Jennimaria Palomaki, Chris Alberti, Daniel Andor, Eunsol Choi, Livio Baldini Soares, and Michael Collins. 2020. QED: A framework and dataset for explanations in question answering. *arXiv:2009.06354*.

Zhenzhong Lan, Mingda Chen, Sebastian Goodman, Kevin Gimpel, Piyush Sharma, and Radu Soricut. 2020. ALBERT: A Lite BERT for self-supervised learning of language representations. In *International Conference on Learning Representations*.

Stephen C. Levinson. 1983. *Pragmatics*, Cambridge Textbooks in Linguistics, pages 181–184, 194. Cambridge University Press.

Zhenghao Liu, Chenyan Xiong, Maosong Sun, and Zhiyuan Liu. 2020. Fine-grained fact verification with kernel graph attention network. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 7342–7351, Online. Association for Computational Linguistics.

Randall Munroe. 2020. Learning new things from Google. https://twitter.com/xkcd/status/1333529967079120896. Accessed: 2021-02-01.

Slav Petrov, Leon Barrett, Romain Thibaux, and Dan Klein. 2006. Learning accurate, compact, and interpretable tree annotation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 433–440, Sydney, Australia. Association for Computational Linguistics.

Pranav Rajpurkar, Robin Jia, and Percy Liang. 2018. Know what you don't know: Unanswerable questions for SQuAD. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 784–789, Melbourne, Australia. Association for Computational Linguistics.

Rob A. Van der Sandt. 1992. Presupposition projection as anaphora resolution. *Journal of Semantics*, 9(4):333–377.

Philippe Schlenker. 2008. Presupposition projection: Explanatory strategies. *Theoretical Linguistics*, 34(3):287–316.

Bernhard Schwarz and Alexandra Simonenko. 2018. Decomposing universal projection in questions. In *Sinn und Bedeutung 22*, volume 22, pages 361–374.

Florian Schwarz. 2016. Experimental work in presupposition and presupposition projection. *Annual Review of Linguistics*, 2(1):273–292.

Mandy Simons. 2013. Presupposing. *Pragmatics of Speech Actions*, pages 143–172.

Peter F. Strawson. 1950. On referring. *Mind*, 59(235):320–344.

Nadine Theiler. 2020. An epistemic bridge for presupposition projection in questions. In *Semantics and Linguistic Theory*, volume 30, pages 252–272.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. 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 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

Galina Tremper and Anette Frank. 2011. Extending fine-grained semantic relation classification to presupposition relations between verbs. *Bochumer Linguistische Arbeitsberichte*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel R. Bowman. 2019. GLUE: A multi-task benchmark and analysis platform for natural language understanding. In *International Conference on Learning Representations*.

Adina Williams, Nikita Nangia, and Samuel Bowman. 2018. A broad-coverage challenge corpus for sentence understanding through inference. 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 1112–1122, New Orleans, Louisiana. Association for Computational Linguistics.

Mark Yatskar. 2019. A qualitative comparison of CoQA, SQuAD 2.0 and QuAC. 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 2318–2323, Minneapolis, Minnesota. Association for Computational Linguistics.

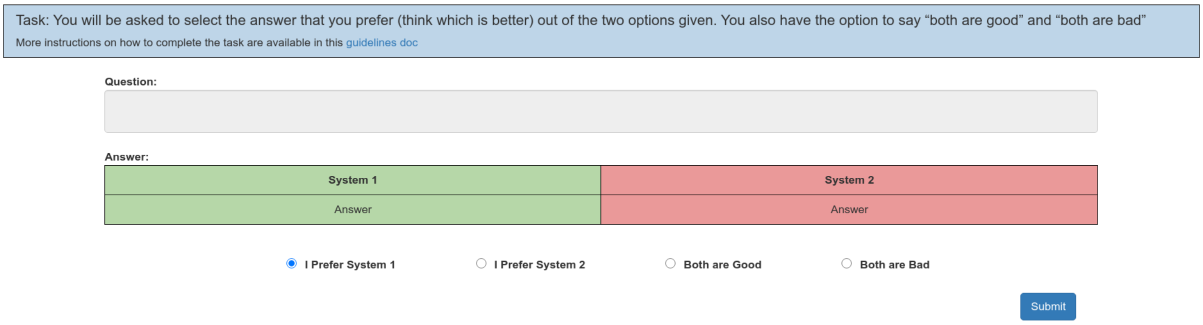
Zhuosheng Zhang, Junjie Yang, and Hai Zhao. 2020. Retrospective reader for machine reading comprehension. *arXiv:2001.09694*.

Haichao Zhu, Li Dong, Furu Wei, Wenhui Wang, Bing Qin, and Ting Liu. 2019. Learning to ask unanswerable questions for machine reading comprehension. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 4238–4248, Florence, Italy. Association for Computational Linguistics.

A Additional Causes of Unanswerable Questions

Listed below are cases of unanswerable questions for which presupposition failure may be partially useful:

- **Document retrieval failure:** The retrieved document is unrelated to the question, so the presuppositions of the questions are unlikely to be verifiable from the document.
- **Failure of commonsensical presuppositions:** The document does not directly support the presupposition but the presupposition is commonsensical.
- **Presuppositions involving subjective judgments:** verification of the presupposition requires subjective judgment, such as the existence of *the best song*.



Task: You will be asked to select the answer that you prefer (think which is better) out of the two options given. You also have the option to say "both are good" and "both are bad"

More instructions on how to complete the task are available in this guidelines doc

Figure 3: The user interface for the user preference study.

- **Reference resolution failure:** the question contains an unresolved reference such as a pro-form (*I, here...*) or a temporal expression (*next year...*). Therefore the presuppositions also fail due to unresolved reference.

B User Study

B.1 Task Design

Figure 3 shows the user interface (UI) for the study. The raters were given a guideline that instructed them to select the answer that they preferred, imagining a situation in which they have entered the given question to two different QA systems. To avoid biasing the participants towards any answer type, we used a completely unrelated, nonsensical example (Q: *Are potatoes fruit?* System 1: *Yes, because they are not vegetables.* System 2: *Yes, because they are not tomatoes.*) in our guideline document.

B.2 DPR Rewrites

The DPR answers we used in the user study were rewrites of the original outputs. DPR by default returns a paragraph-length Wikipedia passage that contains the short answer to the question. From this default output, we manually extracted the sentence-level context that fully contains the short answer, and repaired the context into a full sentence if the extracted context was a sentence fragment. This was to ensure that all answers compared in the study were well-formed sentences, so that user preference was determined by the content of the sentences rather than their well-formedness.

C Presupposition Generation Templates

See Table 6 for a full list of presupposition triggers and templates used for presupposition generation.

D Data Collection

The user study (Section 4) and data collection of entailment pairs from presuppositions and Wikipedia sentences (Section 5) have been performed by crowdsourcing internally at Google. Details of the user study is in Appendix B. Entailment judgements were elicited from 3 raters for each pair, and majority vote was used to assign a label. Because of class imbalance, all positive labels were kept in the data and negative examples were down-sampled to 5 per document.

E Modeling Details

E.1 Zero-shot NLI

MNLI and QNLI were trained following instructions for fine-tuning on top of ALBERT-xxlarge at `https://github.com/google-research/albert/blob/master/albert_glue_fine_tuning_tutorial.ipynb` with the default settings and parameters.

E.2 KGAT

We used the off-the-shelf model from `https://github.com/thunlp/KernelGAT` (BERT-base).

E.3 ETC models

For all ETC-based models, we used the same model parameter settings as Ainslie et al. (2020) used for NQ, only adjusting the maximum global input length to 300 for the flat models to accommodate the larger set of tokens from presuppositions. Model selection was done by choosing hyperparameter configurations yielding maximum Average F1. Weight lifting was done from BERT-base instead of RoBERTa to keep the augmentation experiments simple. All models had 109M parameters.

All model training was done using the Adam optimizer with hyperparameter sweeps of learning

Question (input)	Template	Presupposition (output)
who sings it’s a hard knock life	there is someone that __	there is someone that *sings it’s a hard knock life*
which philosopher advocated the idea of return to nature	some __	some *philosopher advocated the idea of return to nature*
where do harry potter’s aunt and uncle live	there is some place that __	there is some place that *harry potter’s aunt and uncle live*
what did the treaty of paris do for the US	there is something that __	there is something that *the treaty of paris did for the US*
when was the jury system abolished in india	there is some point in time that __	there is some point in time that *the jury system was abolished in india*
how did orchestra change in the romantic period	__	*orchestra changed in the romantic period*
how did orchestra change in the romantic period	there is some way that __	there is some way that *orchestra changed in the romantic period*
why did jean valjean take care of cosette	__	*jean valjean took care of cosette*
why did jean valjean take care of cosette	there is some reason that __	there is some reason that *jean valjean took care of cosette*
when is the year of the cat in chinese zodiac	__ exists	*‘year of the cat in chinese zodiac’* exists
when is the year of the cat in chinese zodiac	__ is contextually unique	*‘year of the cat in chinese zodiac’* is contextually unique
what do the colors on ecuador’s flag mean	__ has __	*‘ecuador’* has *‘flag’*
when was it discovered that the sun rotates	__	*the sun rotates*
how old was macbeth when he died in the play	__	*he died in the play*
who would have been president if the south won the civil war	it is not true that __	it is not true that *the south won the civil war*

Table 6: Example input-output pairs of our presupposition generator. Text in italics denotes the part taken from the original question, and the plain text is the part from the generation template. All questions are taken from NQ.

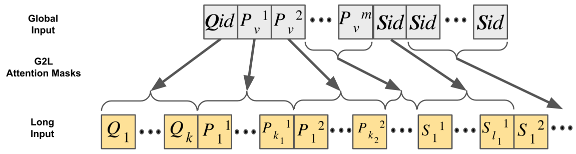


Figure 4: The structured augmentation to the ETC model. Q_k are question tokens, P_k are presupposition tokens, S_l are sentence tokens, P_v are verification labels, Qid is the (constant) global question token and *Sid* is the (constant) global sentence token.

rates in $\{3\times10^{-5}, 5\times10^{-5}\}$ and number of epochs in $\{3, 5\}$ (i.e., 4 settings). In cases of overfitting, an earlier checkpoint of the run with optimal validation performance was picked. All training was done on servers utilizing a Tensor Processing Unit 3.0 architecture. Average runtime of model training with this architecture was 8 hours.

Figure 4 illustrates the structure augmented ETC model that separates question and presupposition tokens that we discussed in Section 6.

F ETC Prediction Change Examples

We present selected examples of model predictions from Section 6 that illustrate the difference in behavior of the baseline ETC model and the structured, presupposition-augmented model:

1. [Correct *Answerable* → *Unanswerable*]
   NQ Question: who played david brent’s girlfriend in the office
   Relevant presupposition: David Brent has a girlfriend
   Wikipedia Article: The Office Christmas specials
   Gold Label: Unanswerable
   Baseline label: Answerable
   Structured model label: Unanswerable
   Explanation: The baseline model incorrectly predicts *arrange a meeting with the second woman (voiced by Julia Davis)* as a long answer and *Julia Davis* as a short answer, inferring that the second woman met by David Brent was his girlfriend. The structured model correctly flips the prediction to *Unanswerable*, possibly making use of the unverifiable presupposition *David Brent has a girlfriend*.

2. [Correct *Unanswerable* → *Answerable*]
   NQ Question: when did cricket go to 6 ball overs
   Relevant presupposition: Cricket went to 6 balls per over at some point
   Wikipedia Article: Over (cricket)
   Gold Label: Answerable
   Baseline label: Unanswerable
   Structured model label: Answerable
   Explanation: The baseline model was likely confused because the long answer candidate only mentions Test Cricket, but support for the presupposition came from the sentence *Although six was the usual number of balls, it was not always the case*, leading the structured model to choose the correct long answer candidate.

3. [Incorrect *Answerable* → *Unanswerable*]
   NQ Question: what is loihi and where does it originate from
   Relevant presupposition: there is some place that it originates from
   Wikipedia Article: Lōihi Seamount
   Gold Label: Answerable
   Baseline label: Answerable
   Structured model label: Unanswerable
   Explanation: The baseline model finds the correct answer (*Hawaii hotspot*) but the struc-

tured model incorrectly changes the prediction. This is likely due to verification error—although the presupposition *there is some place that it originates from* is verifiable, it was incorrectly labeled as unverifiable. Possibly, the the unresolved *it* contributed to this verification error, since our verifier currently does not take the question itself into consideration.