Scoring Multi-hop Question Decomposition Using Masked Language Models

Abdellah Hamouda Sidhoum, Mhamed Mataoui, Faouzi Sebbak, Adil Imad Eddine Hosni, Kamel Smaïli

Scoring Multi-hop Question Decomposition Using Masked Language Models

Abdellah Hamouda Sidhoum *[1], M'hamed Mataoui[1], Faouzi Sebbak[1], Adil Imad Eddine Hosni[1], and Kamel Smaïli[2]

[1] Computer Science Department, Ecole Militaire Polytechnique, Algiers, Algeria
[2] SMarT Group, Loria, University of Lorraine, France

Abstract. We propose a novel approach for multi-hop question decomposition using pre-trained language models scoring in a zero-shot manner. The approach involves generating decomposition candidates, scoring them using a pseudo-log likelihood estimation, and ranking them based on their scores. The evaluation of the decomposition process was carried out using two annotated datasets in two different languages prior to its integration into a complete QA system. We assessed the QA system using the HotpotQA dataset and observed noteworthy results compared to the baseline. The proposed approach highlights the efficiency of the language model scoring technique in complex reasoning tasks like multi-hop question decomposition. Moreover, this technique does not require any training data, making it particularly useful in low-resource language scenarios.

Keywords: Question answering, multi-hop questions, masked language model, sentence scoring.

1 Introduction

Question answering (QA) is a challenging task in Natural language processing (NLP) that measures a system's language understanding and reasoning abilities [22, 18]. Recent advances in Transformer [28] based models, such as BERT [5], T5 [21] and GPT [19, 20, 2, 16], have shown great promise in achieving high performance on the QA task. However, as the complexity of questions increases, more challenging questions are gaining interest in the research community. One such challenge is multi-hop question answering, which requires the system to reason over multiple pieces of evidence to arrive at an answer [32]. Multi-hop question answering can be approached in various ways, such as free text-based, graph-based, and decomposition approaches. In free text-based approaches, the system must identify and extract relevant information from a large corpus of text to answer the question. Graph-based approaches, on the other hand, represent the information as a graph and use graph algorithms to reason over the graph structure. Decomposition approaches break down the question into sub-questions and answer them individually before combining the answers to arrive at the final solution. In this work, we are interested in the multi-hop questions decomposition problem.

The question decomposition technique aims to simplify the multi-hop question into a set of single-hop sub-questions. However, this requires significant amounts of annotated data on decomposition to train generative or extractive models. Recent research has explored the potential of pre-trained language models to overcome limitations associated with using annotated data, particularly in low-resource languages. In light of these developments, our study investigates the effectiveness of pre-trained language models for performing the question decomposition task, without relying on annotated data. We frame the decomposition problem as a ranking problem. Given a question q and a set of decomposition candidates D, each candidate consisting of a set of sub-questions, the task is to determine which candidate provides the most appropriate decomposition of q. The goal is to provide a ranking function $f(q, D)$ that maps each question q to a ranking of the decomposition candidates in D based on their relevance to q. Although the ranking function can be learned using supervised learning techniques , our approach adopts a different strategy by using a 3-step process to rank the decomposition candidates without relying on a learning phase. First, we generate all possible decomposition candidates for each question. Next, we use a pre-trained language model to score these candidates. Finally, we rank the candidates based on these scores to determine the most appropriate decomposition.

Our proposed technique for question decomposition aims to rank decomposition candidates based on the linguistic coherence and acceptability of their sub-questions. Scoring text using masked language models (MLMs) has proven to be an efficient method of evaluating the quality of a sentence. this technique has been used in various natural language processing tasks, such as ranking in Machine Translation and Speech Recognition. One such MLM, BERT, has exceeded previous models by incorporating bidirectional context achieved through masked language modeling (MLM) objectives. MLMs are trained using a self-supervised learning approach to predict the masked tokens based on the context of the surrounding tokens in the text sequence. We leverage the masked language modeling head tuned during the pre-training phase to estimate the scores of the decomposition candidates' sub-questions. We discuss the scoring method further in section 3.3.

Our proposed method is evaluated on two datasets, HotpotQA [32] and an annotated set of questions on extractive decomposition [15], in different settings. The general results show that our approach achieves comparable performance to existing supervised methods. Our main contributions in this paper include the introduction of a novel approach for multi-hop question answering using LM scoring without annotated data, the evaluation of the method on benchmark datasets, and the demonstration of its effectiveness in different settings. Overall, this article aims to contribute to the advancement of multi-hop question answering by proposing a novel approach that leverages pre-trained LMs and does not require annotated data.

2 Related Work

2.1 Multi-hop Reading Comprehension

Multi-hop reading comprehension is a challenging problem in natural language processing, as it requires a model to understand the relationships between multiple pieces of information to answer a question. The most common approaches can be classified into three categories: graph-based, free-text-based, and decomposition approaches. Graph-based approaches leverage the graph structure to represent the text and the question and to model the relationship between them. Examples of such models include CogQA [6], which introduces a cognitive graph for reasoning, and HGN [7], which adopts a hierarchical graph network. DFGN [18] proposes a dynamic fusion graph network to learn the interactions between different parts of the input. Free-text-based approaches treat the passage and question as unstructured text and use techniques such as self-attention to capture the relevant information. Examples include SAE [27], which employs a selective attention mechanism, S2G [31], which uses a graph-based transformer, and FE2H [14], which applies an easy-to-hard learning strategy. Decomposition approaches, as discussed in section 2.2, decompose the original multi-hop question into simpler sub-questions and answer them individually.

2.2 Questions Decomposition

Question decomposition is a natural language processing approach that involves breaking down a complex question into simpler sub-questions that can be easily answered by a single-hop QA model. This approach has been shown to improve reading comprehension performance. Previous work has addressed this challenge through the use of semantic and syntactic analysis or machine learning techniques. The DecompRC model [15] employed a supervised model to decompose the complex question into multiple spans in order to perform the multi-hop QA problem. ONUS [17] trained an unsupervised sequence-to-sequence (seq2seq) model in order to generate sub-questions, using similar matching between complex questions from HotpotQA [32] and simple questions of SQuAD [22] to create some pseudo-data used for the training. Fu et al. [8] propose RERC, a three-stage framework of Relation Extractor, Reader, and Comparator. The relation extractor decomposes the question by extracting the subject and key relations. ALBERT [11] model is used as the Reader. The comparator model is trained to perform various quantitative comparisons and summarize all responses to obtain a conclusive final answer. Guo et al. [9] propose a two-step process that consists of a decomposer model and a reading comprehension model. The decomposer is trained on annotated examples to produce sub-questions. The second step is to train a reading comprehension model based on (question, sub-questions, paragraph, answer) tuples. The authors use LMs such as T5 [21] and Bart [13] as the backbone for the two components of the system. The proposed systems demonstrate the effectiveness of question decomposition in improving the performance of multi-hop QA systems.

2.3 Pre-trained language models for sentence scoring

Sentence scoring aims at measuring the likelihood score of a sentence via a language model (LM). It has been widely used in many natural language processing (NLP) scenarios, such as reranking, which is to select the best sentence from multiple candidates. For instance, it can be used to rerank candidates in neural machine translation (NMT) or automatic speech recognition (ASR) tasks or to evaluate sentences in linguistic acceptability [30]. Recently, large LMs have become the widely used approach for scoring and have been applied in many NLP tasks [25, 26]. Some work [3, 23, 24, 29] have tried to use pre-trained LMs to generate sentence scores on NMT or ASR reranking tasks and achieved some promising results. Lee at al.[12] leverage the perplexity score from LMs as an indicator of unsupported claims for the fact-checking task. In the context of question decomposition, pre-trained LMs can be used to evaluate the quality of the decomposed sub-questions. This allows for the automatic evaluation of decomposition candidates, which can be useful for identifying the most effective ones.

3 Methodology

3.1 Overview

Multi-hop question answering requires the system to reason over multiple pieces of information in order to arrive at an answer, making it a challenging task. One solution to this problem is to decompose the complex question into simpler sub-questions, which can be answered using a single piece of information. This approach can improve the accuracy of the overall system by reducing the complexity of the question.

Previous works on complex question decomposition use annotated data and fine-tuned models to achieve this task. Our proposed approach for multi-hop question decomposition relies solely on the knowledge in pre-trained language models and consists of a three-step process illustrated in Figure 1.

1. The first step involves generating decomposition candidates based on the question reasoning type and applying some linguistic constraints to filter out obviously unacceptable candidates. The method for generating decomposition candidates is described in Section 3.2.
2. Then, The second step entails assigning a score to each sub-question generated using the pseudo-log likelihood estimated from a direct inference on masked language models. This scoring procedure is detailed in Section 3.3.
3. Finally, the third step involves ranking the candidates based on an aggregated score using some aggregation functions. The ranking step is explained in Section 3.4.

Overall, this approach uses pre-trained language models to generate, score, and rank multi-hop question decomposition candidates. The approach has the potential to improve QA systems by better decomposing complex questions into simpler sub-questions, thereby improving the accuracy of the overall system.

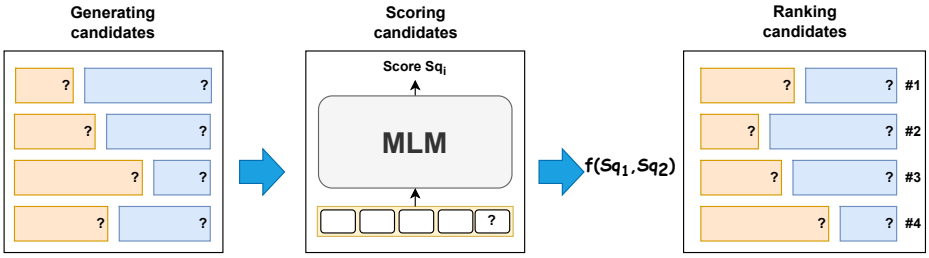

Fig. 1: The proposed decomposition process

3.2 Generation of Decomposition Candidates

According to [15], the authors have identified several reasoning types in multi-hop reading comprehension. These reasoning types are bridge, intersection, and comparison. The authors found that among 184 multi-hop questions out of a sample of 200 from the dev set of hotpotQA [32], 47% were bridge questions, 23% were intersection questions, and 22% were comparison questions. Table 1 provides examples of each reasoning type.

Bridge
Q: what was the population of **the city where penobscot marine museum is located** ?
Sq1: the city where penobscot marine museum is located ?
Sq2: what was the population of [ANSWER] ?
Intersection
Q: 12 years a slave starred **what british actor** born 10 july 1977 ?
Sq1: 12 years a slave starred what british actor ?
Sq2: what british actor born 10 july 1977 ?
Comparison
Q: Were **Scott Derrickson** and **Ed Wood** of the same nationality?
Sq1: Scott Derrickson nationality ?
Sq2: Ed Wood nationality ?
Sq3: is_same(ANS1,ANS2) ?

Table 1: Example of reasoning types of multi-hop questions on hotpotQA

As illustrated in the example, the sub-questions can be extracted as spans from the original multi-hop question, where:

- for bridge type: the first sub-question is a span included in the original question (highlighted in green), and the second sub-question is formed by substituting the span of the first sub-question by a placeholder token ([ANSWER] token in the example) as the answer of the first sub-question.
- for intersection type: there is a shared span between both sub-questions (highlighted in red), and other tokens for each sub-question.

– for comparison type: two compared entities can be extracted (highlighted in yellow) and an operation over some properties of the entities.

We generate decomposition candidates for a given question $Q = \{w_1, .., w_n\}$ based on the reasoning type following Algorithm 1 inspired by the algorithm proposed by Min et al. [15].

Algorithm 1 Generate decomposition candidates

Require: Q: question
Ensure: Decomposition_candidates
/*For bridge type*/:
$Q\_tokens \leftarrow tokenize(Q)$
$L \leftarrow length(Q\_tokens)$
for i **in** $[0, L-1]$ **do**
  for j **in** $[i, L-1]$ **do**
    $Sq_1 \leftarrow Q\_tokens\ [i{:}j+1]$
    $Sq_2 \leftarrow Q\_tokens\ [:i]$+[ANSWER]+ $Q\_tokens\ [j+1:]$
    $candidate \leftarrow [Sq_1, Sq_2]$
    $Decomposition\_candidates.add(candidate)$
  end for
end for
/*For intersection type*/:
for i **in** $[0, L-1]$ **do**
  for j **in** $[i, L-1]$ **do**
    $Sq_1 \leftarrow Q\_tokens\ [:j]$
    $Sq_2 \leftarrow Q\_tokens\ [i+1:]$
    $candidate \leftarrow [Sq_1, Sq_2]$
    $Decomposition\_candidates.add(candidate)$
  end for
end for
/*For Comparison type, we reuse the DecompRC predictions*/:
return $decomposition\_candidates$

We explain below the generation algorithm for each reasoning type:

Bridging Type Sub-Questions: To generate bridge type sub-questions, we need to select a subset of words from the original question as the first sub-question Sq_1:

$$Sq_1 = \{w_i, ..., w_j\} \tag{1}$$

Whereas the second sub-question is formed using the remaining set of words with the keyword $[ANSWER]$ substituting the first subset as follows:

$$Sq_2 = \{w_1, ..., w_{i-1}\} \cup [answer] \cup \{w_{j+1}, ..., w_n\} \tag{2}$$

Intersection Type Sub-Questions: To generate intersection type sub-questions, we need to identify three parts from the original question, two different

subsets of words S_1, S_3, and a shared subset of words S_2 as follows:

$$S_1 = \{w_1, ..., w_{i-1}\} \tag{3}$$

$$S_2 = \{w_i, ..., w_j\} \tag{4}$$

$$S_3 = \{w_{j+1}, ..., w_n\} \tag{5}$$

Then, the two intersection type sub-questions can be generated as:

$$Sq_1 = S_1 \cup S_2 \tag{6}$$

$$Sq_2 = S_2 \cup S_3 \tag{7}$$

In order to generate valid sub-questions from a given question, several constraints are typically applied. Our hypothesis suggests that these conditions include:

- Length: A valid question should be of an appropriate length, typically consisting of at least three words. This ensures that the question has enough content to convey a meaningful message.
- Stopwords: Stopwords are common words that do not carry much meaning on their own, such as "the," "a," "of," "and," etc. A valid question should contain at least one word that is not a stopword, which ensures that the question has some meaningful content.
- Grouping of Noun Phrase Tokens: Another condition involves grouping noun phrase tokens together, which can help to create more semantically meaningful sub-questions. By combining noun phrase tokens into a single sub-question, we ensure that tokens with relation to the same entity are present together on the same sub-question.

Noun phrases are typically composed of a noun and other words that modify or describe the noun. To extract noun phrases from a sentence, we can use Part-of-Speech (POS) tagging to identify the nouns and other parts of speech that typically appear in noun phrases, such as determiners, adjectives, and pronouns. Once these parts of speech are identified, we can group them together with the noun they modify to form a noun phrase. This can be done using Algorithm 2 provided. The figure 2 shows an example of a multi-hop question tokenized and tagged using nltk[3] with noun phrases grouped. As illustrated, tokens such as "New York city" must always be together on the same group which constitute a meaningful entity.

3.3 Masked Language Model Scoring

This section introduces the scoring methods that adopt the masked word prediction of MLMs. These methods aim to assign a higher likelihood to sub-questions that are both syntactically and semantically correct while assigning a lower likelihood to sub-questions that are incoherent, incorrect, or infrequently used. The

[3] The natural language toolkit (NLTK) https://www.nltk.org/

Algorithm 2 Question tokenization with noun phrase tokens grouping

Require: a question Q
Ensure: question_tokens_grouped
tagged_tokens ← *pos_tag*(Q)
$Noun\_Phrase\_tags \leftarrow [RB, DT, JJ, JJS, NN, NNS, NNP, NNPS, PRP]$
for *token*, *pos* **in** *tagged_tokens* **do**:
    if *pos* **in** $Noun\_Phrase\_tags$ **then**:
        append *token* to *current_noun_phrase*
    else
        if current_noun_phrase **is** not empty **then**:
            append *current_noun_phrase* to *question_tokens_grouped*
            current_noun_phrase ← [ ]
        end if
        append *token* to *question_tokens_grouped*
    end if
end for
return *question_tokens_grouped*

Question: The director of the romantic comedy "Big Stone Gap" is based in what New York City?
Question tokens with tags: [(The, DT), (director, NN), (of, IN), (the, DT), (romantic, JJ), (comedy, NN), (Big, NNP), (Stone, NNP), (Gap, NNP), (is, VBZ), (based, VBN), ('in', 'IN'), ('what', WP), (New, NNP), (York, NNP), (city, NN), (?, .)]
Question tokens with noun phrases grouped:[The director, of, the romantic comedy Big Stone Gap, is, based, in, what, New York city, ?]

Fig. 2: An example of a multi-hop question tokenized and tagged using nltk with noun phrases grouped.

basic principle of sentence scoring using MLMs is to mask a word in a given sentence and then compute the probability of the original word on the masked position. Finally, the score of the given sentence is obtained by summing all log-likelihoods of the masked words from each input instance of the sentence changing the position of the masked token. Figure 3 shows the masking mechanism in different positions in a sentence and the estimation of their likelihood probabilities using the conditional probabilities induced by applying the softmax function on top of a Language model pre-trained on masked word prediction objective.

This obtained score is known as the pseudo-log likelihood score (PLLs). In a recent study by Salazar et al. [23], PLLs from MLMs were examined. The PLL score is calculated using the following formula:

$$PLL(W) = \sum_{t=1}^{|W|} logP(w_t \mid W_{\setminus w_t}; \Theta) \tag{8}$$

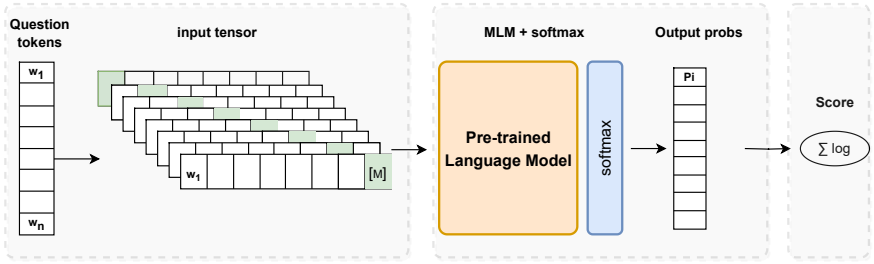

Fig. 3: Masked language model scoring

where $W_{\setminus w_t} = (w_1, .., w_{t-1}, [MASK], w_{t+1}, .., w_{|W|})$ denote the sentence W with the word w_t replaced by a special token [MASK]. $P(w_t \mid W_{\setminus w_t})$ is the conditional probabilities of each sentence token induced in the MLM given its bidirectional context $W_{\setminus w_t}$, and θ denotes the model's parameters.

Analogous to LMs, Salazar et al. propose the *pseudo-perplexity*(PPPL) as a measure of bidirectional language model performance in modeling a corpus of sentences. The pseudo-perplexity is computed using the equation (9), where $\mathbb{C}$ denotes the corpus and N is the number of tokens in the corpus.

$$PPPL(\mathbb{C}) = exp\left(-\frac{1}{N}\sum_{W\in\mathbb{C}} PLL(W)\right) \tag{9}$$

According to the authors, both PLLs and PPPLs are scores that we can assign to sentences and corpora and have been adopted to evaluate generated text quality. In our context of question decomposition, we put forward two hypotheses while scoring using PPPLs:

- Hypothesis 1 (H1): $\mathbb{C} = Sq_i$, the score is calculated for each sub-question of the candidate. Hense, The scoring formula (9) is simplified as follow:

$$PPPL(W) = exp\left(-\frac{1}{|W|}PLL(W)\right) \tag{10}$$

  In H1, two scores are assigned for each candidate. The formula obtained is a normalized form of PLL, as the score is divided by the number of tokens present in the sentence $|W|$. Duo to the linear correlation between PLLs and $|W|$[23], employing a length-wise averaging enables meaningful comparison between scores of sentences with different lengths.
- Hypothesis 2 (H2): $\mathbb{C} = \{Sq_1, Sq_2\}$, the corpus for this case is the set of sub-questions for each decomposition candidate. Only one score is assigned for each candidate.

We propose two additional scoring settings concerning the sub-questions, in order to see their effect on the scoring phase:

– Concatenation: We propose to concatenate the two sub-questions before scoring. The input to the scoring module will be: $Sq_1 + "." + Sq_2$. We obtain one score for the candidate.
– Question form: Adding a question word (Q-word) to the beginning of the sub-question (if not exist) and a question mark at the end may help to differentiate between well-formed sub-questions and others while scoring.

We investigate the aforementioned scoring methods through experiments, which are detailed in section 4.3.

3.4 Ranking of decomposition candidates

The problem of ranking decomposition candidates based on their scores arises when each candidate has two scores, s_1 and s_2, referring to the scores of sub-question 1 and sub-question 2, respectively. This issue is not present when scoring concatenated sub-questions and in H1 (scoring using PPPL with $\mathbb{C} = \{Sq_1, Sq_2\}$) since only one score for each candidate is present. However, To rank the decomposition candidates scored using PLL and in H2, it is necessary to combine these two scores into a single score.

To aggregate the scores, different aggregation functions could be used. One intuitive approach is to sum the two scores. However, simply adding the two scores together may not be appropriate, as one sub-question may have a much higher score than the other, which could bias the final score towards that sub-question. Another alternative is to use an aggregate function that considers the difference between the two scores. This could be achieved, for example, by subtracting the absolute difference between the two scores from their sum as presented in equation (11).

$$candidate\_score = (s_1 + s_2) - |s_1 - s_2| \tag{11}$$

The objective of this formula is to encourage equitable contributions from both sub-questions, mitigating instances where one sub-question significantly biases the final score. By penalizing large score disparities, we ensure that both sub-questions play a substantial and balanced role in determining the aggregate score.

To illustrate this issue, we provide the following example: *Question: What was the real name of the star of the 1963 film 'The Nutty Professor' ?*. A subset of the generated candidates and their PLL scores are presented in Table 2.

In this example, candidate (4) has a high sum score due to its first sub-question having a high score, even though its second sub-question has a low score. This can make candidate (4) appear to be the best decomposition candidate. However, when using the sum-difference as the aggregated score, this issue is mitigated to some extent, as the sum-difference score takes into account the difference between the scores of the two sub-questions. In this case, candidate (2) would be ranked higher than candidate (4), since candidate (2) has more balanced scores for both sub-questions and a more minor difference between the scores.

#	Decomposition candidates	PLL	Sum	Sum-Diff
1	SQ1: of the star of the 1963 film 'the Nutty Professor' SQ2: What was the real name [ANSWER] ?	-36.77 -27.68	-64.45	-73.54
2	SQ1: the star of the 1963 film 'the nutty professor' SQ2: What was the real name of [ANSWER] ?	-26.70 -26.56	-53.26	-53.4
3	SQ1: the 1963 film the Nutty Professor SQ2: What was the real name of the star of [ANSWER] ?	-22.58 -33.51	-56.09	-67.02
4	SQ1: the real name of the star SQ2: What was [ANSWER] of the 1963 film the Nutty Professor ?	-16.26 -27.89	-44.11	-55.74

Table 2: Example of decomposition candidates generated and their scores

Expanding upon these propositions, we introduce weighted forms of the initial aggregation functions to accommodate varying importance levels for sub-questions. In the "weighted sum" aggregation (see Equation (12)), a parameter α is introduced. This parameter influences the balance between the contributions of the two sub-questions to the final score.

$$Sc_{Wsum} = \alpha * s_1 + (1 - \alpha) * s_2 \tag{12}$$

Finally, our "weighted sum-diff" aggregation (Equation (13)) leverages the parameter α to ensure a balance between sub-questions influences on the final score, while also considering their absolute score differences.

$$Sc_{Wsum-diff} = \alpha * (S_1 + S_2) - (1 - \alpha) * |S_1 - S_2| \tag{13}$$

With $\alpha = 0.5$, we are expressing the initial aggregation functions. The proposed functions for aggregating the scores of the decomposition candidates were experimentally evaluated, in order to select the appropriate function for our purpose and to verify our hypothesis. The details of these experiments will be presented in the experiments section.

3.5 Single-hop Reading Comprehension

The step after question decomposition is Reading Comprehension, in which we leverage an off-the-shelf single-hop QA model to answer the sub-questions. In this work, we integrate our approach into a complete QA pipeline in order to assess the quality of the decomposition component in the global QA system. To this end, we rely on the DecompRC [15] QA models and substitute its decomposition predictions with our predictions.

The DecompRC system uses the paragraph selection approach from Clark and Gardner [4] to support multiple paragraphs and BERT [5] as a reading comprehension model. The system uses single-hop questions obtained from SQuAD [22] and easy examples of hotpotQA [32] to form the training data. Three instances with different settings were trained for an ensemble to be used as the single-hop RC model.

Once the reading comprehension step has been completed, the decomposition candidates that are generated include sub-questions, their respective answers, and supporting evidence that relates to the specific reasoning type. DecompRC employs a decomposition scorer to determine the most appropriate decomposition and ultimately selects the answer from that decomposition as the final answer. The scoring process involves encoding a concatenated sequence of words that comprises the original question, the reasoning type, the answer, and the evidence using BERT. The resulting encoding is then fed through a trainable matrix to obtain a score that indicates whether the decomposition leads to a correct final answer. During inference, DecompRC takes the reasoning type that leads to the highest score and the answer corresponding to that type is the final answer.

4 Experiments

The quality of the question decomposition phase is a critical factor in ensuring the overall effectiveness of the QA system, as it directly impacts the accuracy of the final answer. Evaluating the quality of the decomposition phase can help us identify areas for improvement in our approach, make informed decisions about optimizing the system, and compare the effectiveness of different decomposition strategies or models. Additionally, it allows us to determine which configuration is most effective for what type of questions, dataset, and language.

4.1 Datasets

In this section, we present the different datasets used in our experiments for evaluating the decomposition phase and subsequently assessing reading comprehension (RC) within a complete QA system. For decomposition evaluation, We include datasets in two languages, English and Arabic, to assess our proposition's adaptability and generalization across languages.

- $\mathcal{D}_1$**: HotpotQA.** The HotpotQA dataset [32] is a collection of multi-hop questions and their corresponding answers, specifically designed to test a machine's ability to reason over multi-hop relationships in given passages to find the answer. It consists of over 113k questions, sourced from Wikipedia articles by crowdworkers. The questions in the HotpotQA dataset are categorized into two types: bridge and comparison (different from those by [15] in Table 1). We evaluate the RC on the dev set that contains 7405 questions.
- $\mathcal{D}_2$**: HotpotQA annotated dataset for decomposition.** To evaluate the decomposition phase, we utilized a subset of questions from $\mathcal{D}_1$ that were annotated for extractive decomposition. The constructed dataset was used to train the pointer model[15], which maps the question to indices that define the span of the sub-questions. Each question is annotated on decomposition by adding special marks indicating the start and end positions of sub-questions (letter **S** in Figure 4). The dataset comprised a total of 618 questions, with 352 belonging to the bridge type and 266 belonging to the intersection type. From the annotated questions, we create sub-questions to be used as reference decomposition while comparing them with the predicted sub-questions.

Question: What was the population of **S** the city where penobscot marine museum is located **S** ?

SQ1: the city where penobscot marine museum is located.
SQ2: What was the population of [ANSWER]?

Fig. 4: An example of a question from hotpotQA annotated for question decomposition.

- $\mathcal{D}_3$**: ACQAD** Arabic Complex Question Answering Dataset[10][4] is a collection of multi-hop questions in Arabic language. The dataset contains 115k questions of comparison type and 2984 questions of bridging type. The questions were generated automatically from Wikipedia. Each question-answer pair is provided with its decomposition into single-hop sub-questions with their corresponding answers. In this work, we focused only on the bridging type. Figure 5 provides an example from the dataset. We made a modification to the second sub-question (SQ2) in each question decomposition to ensure consistency with extractive decomposition. We replaced the answer to SQ1 (in the example provided is: Britain) with a placeholder token [UNK] that could be used to handle unseen tokens (Unknown answer in our case).

Question:
كم يبلغ سن التقاعد في البلد الذي استضاف الألعاب الأولمبية الصيفية لعام ١٩٤٨ ؟
What is the retirement age in the country that hosted the 1948 Summer Olympics?

SQ1: ما هو البلد الذي استضاف الألعاب الأولمبية الصيفية لعام ١٩٤٨ ؟.
What country hosted the 1948 Summer Olympics?

SQ2: كم يبلغ سن التقاعد في بريطانيا ؟
What is the retirement age in Britain?

Fig. 5: An example of a question from ACQAD annotated for question decomposition.

4.2 Experimental setup

– **Scoring Models** We use the Bert-base-uncased model from the GluonNLP toolkit[5] as our Masking language model for scoring in English. We utilize Salazar's implementation of PLL and PPPL from the GitHub repository[6].

[4] The dataset is available on Kaggle: https://www.kaggle.com/datasets/abdellahhamouda/acqad-dataset
[5] https://nlp.gluon.ai/
[6] https://github.com/awslabs/mlm-scoring

For the Arabic language, we employ the bert-base-arabertv2 model [1] from the HuggingFace Transformers library [7].

- **Evaluation metrics for decomposition** To evaluate the quality of the generated sub-questions in the question decomposition task, we chose the BLEU score as a commonly used metric for evaluating the quality of a machine-generated text. The BLEU score is well-suited for this evaluation, as it measures n-gram overlap between the generated sub-questions and the reference sub-questions [9]. By using the BLEU score, we can obtain an objective measure of the effectiveness of our decomposition approach.
- **Reading Comprehension models** We utilized an off-the-shelf single-hop RC model from DecompRC. The training details were described in section 3.5. We relied on all the pre-trained models and data from the DecompRC repository[8]. The architecture of this single-hop RC model is built upon Hugging Face's bert-base-uncased implementation[9]. This architecture comprises 12 layers of Transformers and operates with a hidden dimension of 768.

4.3 Evaluation results for the decomposition approach

We conducted a series of experiments on $\mathcal{D}_2$ and $\mathcal{D}_3$ to investigate three key aspects of our study: the scoring formulas, the aggregation functions, and question types.

Results on $\mathcal{D}_2$ The results of our experiments are presented in Figure 6. It depicts the BLEU scores while scoring with PLL compared to PPPL(with Hypothesis 1). The results are displayed with respect to each question type and aggregation function, for a range of α values.

The obtained results indicate several noteworthy trends:

- **Scoring Formulas:** Comparing the two scoring formulas, it is notable that scoring using PPPL consistently outperforms PLL. Across all question types and aggregation functions.
- **Aggregation Functions:** In terms of aggregation functions, using the sum function consistently leads to higher BLEU scores compared to the sum-diff function. This pattern holds true for both scoring formulas (PPPL and PLL) and all question types.
- **Effect of α:** While analyzing the effect of different α values, varying α does influence the BLEU scores. However, the consistent trend of PPPL + sum yielding higher scores remain consistent across varying α values.

Results on $\mathcal{D}_3$ The outcomes of our experiments on $\mathcal{D}_3$ are presented in Figure 7. We compare the performance of PLL and PPPL(with hypothesis 1) and display the results for each aggregation function separately. In contrast to $\mathcal{D}_2$, PLL generates higher BLEU scores compared to PPPL, Whereas aggregating with sum still performs better than sum-diff. Note that the PLL and PPPL scores are based heavily on what text the model was trained on. This means that these results are not comparable between models or datasets in different languages.

[7] https://huggingface.co/aubmindlab/bert-base-arabertv2
[8] https://github.com/shmsw25/DecompRC
[9] https://github.com/huggingface/ pytorch-pretrained-BERT

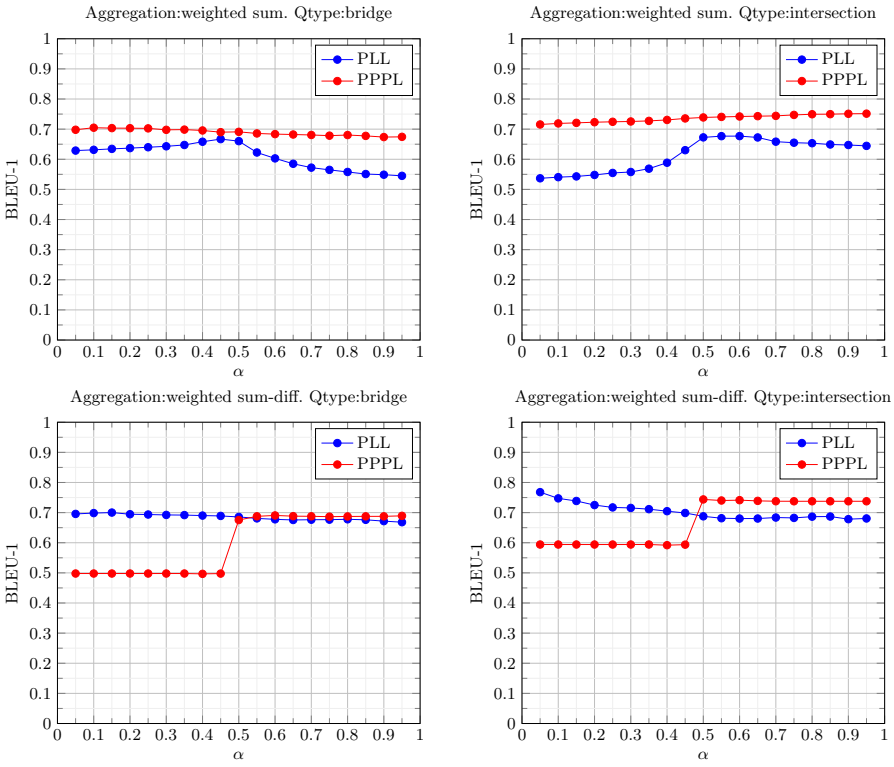

Fig. 6: Evaluation results for $\mathcal{D}_2$

Concatenation and Question form effect We conducted experiments to evaluate the impact of scoring after concatenating sub-questions and scoring after adding a question word to the generated sub-questions.

The evaluation results, as shown in Figure 8, present a comparison of BLEU scores before and after the addition of a question word to the generated sub-questions. We relied on the optimal configuration from the preceding experiments, i.e. PPPL + sum for $\mathcal{D}_2$ and PLL + sum for $\mathcal{D}_3$. Subsequently, we introduce the question word setting to the evaluation. We observe that the addition of the question word influences the overall performances for all question types and datasets. It exhibits an improvement in BLEU score by up to 1% for bridge type and 2.5% for intersection type in $\mathcal{D}_2$ and up to 9% for $\mathcal{D}_3$.

Tables 3 and 4 provide an overview of the experimental results across different configurations. We consider the best combination (scoring formula, aggregation function,α value) from the previous experiments as the base configuration (At the first row of both tables). Furthermore, the tables present the results of the concatenation, the introduction of a question word, and the scoring with PPPL using hypothesis 2(see section 3.3).

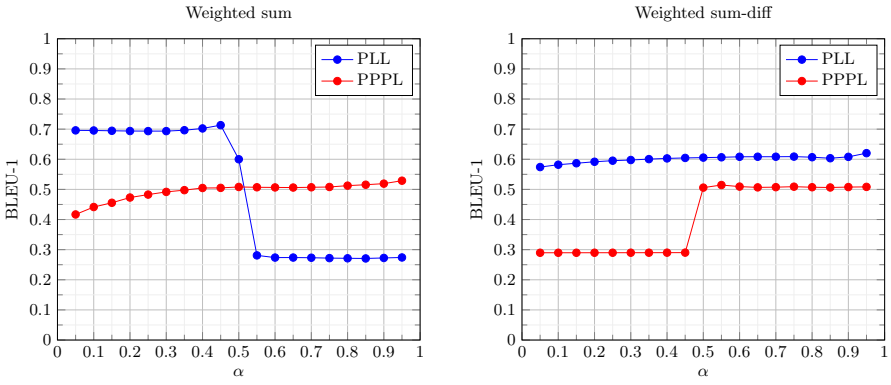

Fig. 7: Evaluation results for $\mathcal{D}_3$

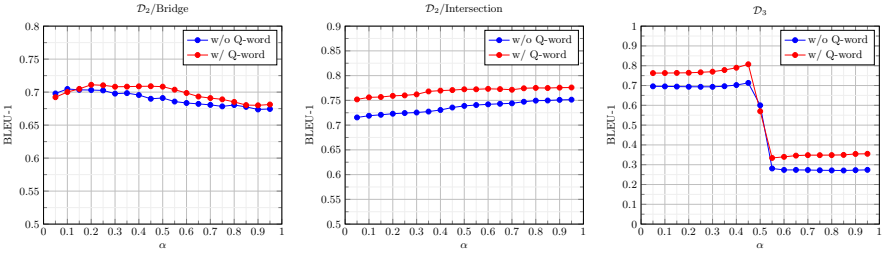

Fig. 8: Evaluation results without and with a question word

.

– Concatenation: We observe that the concatenation strategy dramatically reduces the performance. For instance, in $\mathcal{D}_2$, the BLEU score for Intersection questions decreases from 75.14 to 65.4 losing 10% BLEU score.

– Scoring using PPPL with Hypothesis 2: Our examination of the PPPL scoring strategy with Hypothesis 2 reveals that this strategy does not exhibit strong performance when compared to alternative scoring configurations for both datasets. However, its impact is amplified when combined with the Q-word. we observe a remarkable boost in performance, with BLEU scores of 78.08% for the Intersection question type surpassing all other configurations.

– Adding Q-word: The addition of a question word consistently results in performance improvements across all configurations and datasets. For instance, in $\mathcal{D}_2$, the inclusion of a question word elevates the BLEU score for Bridge-type questions to 71.11%. In the case of Dataset $\mathcal{D}_3$, this effect is even more pronounced, as the BLEU score reaches 80.73%. These findings affirm the consistent and significant impact of the question form in augmenting the scoring quality.

Configuration	Question Type	
	Bridge	Intersection
PPPL(H1) + sum	70.31	75.14
+ Q-word	**71.11**	77.60
PPPL(H1) + Concat	65.81	65.42
+ Q-word	66.58	72.47
PPPL (H2)	69.45	73.65
+ Q-word	70.64	**78.08**

Table 3: Best results for $\mathcal{D}_2$

Configuration	BLEU
PLL + sum	71.32
+ Q-word	**80.73**
PLL + Concat	64.63
+ Q-word	74.31
PPPL (H2)	56.98
+ Q-word	63.37

Table 4: Best results for $\mathcal{D}_3$

4.4 Comparison between decomposition models

In order to compare between supervised and unsupervised decomposition approaches, we first established a baseline for our evaluation. In this baseline, we implemented a random ranking of generated decomposition candidates. For the supervised method, we evaluate the decomposer model from DecompRC[15] on a test subset from $\mathcal{D}_2$ and measured the BLEU score. Table 5 presents the obtained details. Notably, the proposed approach indicates a significant performance over the baseline. These findings demonstrate the effectiveness of scoring using MLM in identifying good decomposition candidates and ranking them first. Furthermore, when compared to the supervised DecompRC model, our unsupervised method performs competitively, particularly for Intersection-type questions in $\mathcal{D}_2$.

Dataset		Decomposition Model		
		Random Ranking	DecompRC (supervised)	MLMS (unsupervised)
$\mathcal{D}_2$	Bridge	48.58	87.03	71.11
	Intersection	54.07	81.16	78.08
$\mathcal{D}_3$		44.43	/	80.73

Table 5: Comparison between decomposition models

Table 6 shows a comparison between the decomposition predictions generated by our proposed method and the DecompRC decomposer. Notably, our approach was not trained on any annotated data, while the DecompRC decomposer was trained on a dataset annotated on extractive decomposition. Despite this difference, our proposed method gives better predictions than the DecompRC decomposer in some examples. These results suggest that the use of pre-trained

language models for scoring decomposition candidates can be highly effective for solving complex reasoning tasks, even without access to large annotated datasets.

Question	Predicted decompositions
(1) The king of the Molossians and Epirus encamped between what two cities?	**Our:** - The king of the Molossians and Epirus - [ANSWER] encamped between what two cities ? **DecompRC:** - The king of the Molossians and Epirus encamped - [ANSWER] between what two cities ?
(2) What Hong Kong actor born in 1980 will star in "Heart and Greed"?	**our:** - actor will star in "Heart and Greed"? - What Hong Kong actor born in 1980 ? **DecompRC;** - What Hong Kong actor born in 1980 will star ? - What Hong Kong actor born in "Heart and Greed"?

Table 6: Example of decomposition predicted by our method versus those of decompRC decomposer

4.5 Question answering system evaluation

The performance of our proposed approach integrated with the DecompRC pipeline, was compared with BiDAF as the baseline on HotpotQA dataset and DecompRC with random ranking on the HotpotQA development set. The results in Table 5 show the F1 scores achieved by each model. It was observed that the proposed approach outperformed the baseline by a margin of 7%. This demonstrates the effectiveness of the proposed approach in improving the performance of the QA system for the multi-hop question decomposition task.

Model	F1		
	All	**Bridge**	**Comp**
BiDAF (baseline)	58.28	59.09	55.05
DecompRC with Random ranking	60.86	60.39	62.70
DecompRC with PPPL(H2)	66.02	66.87	62.65
DecompRC with PPPL+sum	**67.15**	**67.58**	62.35

Table 7: F1 score on the dev set of hotpotQA in distractor setting

5 Conclusion and future work

In this research, we proposed a novel approach for the multi-hop question decomposition task using pre-trained language models scoring in a zero-shot manner.

Our approach consists of three main steps that involve generating decomposition candidates, scoring the candidates using a pseudo-log likelihood estimation, and ranking the candidates based on an aggregated score. We evaluated our approach on an annotated dataset on question decomposition, in order to experimentally verify our hypotheses on the decomposition process before integrating it into a complete QA system. Moreover, the evaluation of the complete QA pipeline on the HotpotQA dev set showed significant improvements over the baseline. The proposed approach demonstrated the effectiveness of applying the language model scoring techniques in complex reasoning tasks like multi-hop question decomposition. Future work in this area may focus on further improving the accuracy of the decomposition step, exploring the use of other pre-trained language models, and applying the proposed approach to other QA datasets. Additionally, more research can be done to evaluate the proposed approach in more diverse and complex scenarios.

References

1. Antoun, W., Baly, F., Hajj, H.: Arabert: Transformer-based model for arabic language understanding. In: Proceedings of the 4th Workshop on Open-Source Arabic Corpora and Processing Tools, with a Shared Task on Offensive Language Detection. pp. 9–15 (2020)
2. Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J.D., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., et al.: Language models are few-shot learners. Advances in neural information processing systems **33**, 1877–1901 (2020)
3. Chiu, S.H., Chen, B.: Innovative bert-based reranking language models for speech recognition. In: 2021 IEEE Spoken Language Technology Workshop (SLT). pp. 266–271. IEEE (2021)
4. Clark, C., Gardner, M.: Simple and effective multi-paragraph reading comprehension. arXiv preprint arXiv:1710.10723 (2017)
5. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805 (2018)
6. Ding, M., Zhou, C., Chen, Q., Yang, H., Tang, J.: Cognitive graph for multi-hop reading comprehension at scale. arXiv preprint arXiv:1905.05460 (2019)
7. Fang, Y., Sun, S., Gan, Z., Pillai, R., Wang, S., Liu, J.: Hierarchical graph network for multi-hop question answering. arXiv preprint arXiv:1911.03631 (2019)
8. Fu, R., Wang, H., Zhang, X., Zhou, J., Yan, Y.: Decomposing complex questions makes multi-hop qa easier and more interpretable. arXiv preprint arXiv:2110.13472 (2021)
9. Guo, X.Y., Li, Y.F., Haffari, G.: Complex reading comprehension through question decomposition. In: Proceedings of the The 20th Annual Workshop of the Australasian Language Technology Association. pp. 31–40 (2022)
10. Hamouda Sidhoum, A., Mataoui, M., Sebbak, F., Smaïli, K.: Acqad: A dataset for arabic complex question answering. In: International Conference on Cyber Security, Artificial Intelligence and Theoretical Computer Science (2022)
11. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: Albert: A lite bert for self-supervised learning of language representations. arXiv preprint arXiv:1909.11942 (2019)

12. Lee, N., Bang, Y., Madotto, A., Fung, P.: Towards few-shot fact-checking via perplexity. In: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 1971–1981 (2021)
13. Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., Zettlemoyer, L.: Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. arXiv preprint arXiv:1910.13461 (2019)
14. Li, X.Y., Lei, W.J., Yang, Y.B.: From easy to hard: Two-stage selector and reader for multi-hop question answering. arXiv preprint arXiv:2205.11729 (2022)
15. Min, S., Zhong, V., Zettlemoyer, L., Hajishirzi, H.: Multi-hop reading comprehension through question decomposition and rescoring. arXiv preprint arXiv:1906.02916 (2019)
16. OpenAI*: Gpt-4 technical report. arXiv preprint arXiv:2303.08774 (2023)
17. Perez, E., Lewis, P., Yih, W.t., Cho, K., Kiela, D.: Unsupervised question decomposition for question answering. arXiv preprint arXiv:2002.09758 (2020)
18. Qiu, L., Xiao, Y., Qu, Y., Zhou, H., Li, L., Zhang, W., Yu, Y.: Dynamically fused graph network for multi-hop reasoning. In: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics. pp. 6140–6150 (2019)
19. Radford, A., Narasimhan, K., Salimans, T., Sutskever, I., et al.: Improving language understanding by generative pre-training (2018)
20. Radford, A., Wu, J., Child, R., Luan, D., Amodei, D., Sutskever, I., et al.: Language models are unsupervised multitask learners. OpenAI blog **1**(8), 9 (2019)
21. Raffel, C., Shazeer, N., Roberts, A., Lee, K., Narang, S., Matena, M., Zhou, Y., Li, W., Liu, P.J.: Exploring the limits of transfer learning with a unified text-to-text transformer. The Journal of Machine Learning Research **21**(1), 5485–5551 (2020)
22. Rajpurkar, P., Zhang, J., Lopyrev, K., Liang, P.: Squad: 100,000+ questions for machine comprehension of text. arXiv preprint arXiv:1606.05250 (2016)
23. Salazar, J., Liang, D., Nguyen, T.Q., Kirchhoff, K.: Masked language model scoring. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 2699–2712 (2020)
24. Shin, J., Lee, Y., Jung, K.: Effective sentence scoring method using bert for speech recognition. In: Asian Conference on Machine Learning. pp. 1081–1093. PMLR (2019)
25. Song, K., Leng, Y., Tan, X., Zou, Y., Qin, T., Li, D.: Transcormer: Transformer for sentence scoring with sliding language modeling. arXiv preprint arXiv:2205.12986 (2022)
26. Tamborrino, A., Pellicano, N., Pannier, B., Voitot, P., Naudin, L.: Pre-training is (almost) all you need: An application to commonsense reasoning. arXiv preprint arXiv:2004.14074 (2020)
27. Tu, M., Huang, K., Wang, G., Huang, J., He, X., Zhou, B.: Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In: Proceedings of the AAAI conference on artificial intelligence. vol. 34, pp. 9073–9080 (2020)
28. Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A.N., Kaiser, ., Polosukhin, I.: Attention is all you need. Advances in neural information processing systems **30** (2017)
29. Wang, A., Cho, K.: Bert has a mouth, and it must speak: Bert as a markov random field language model. arXiv preprint arXiv:1902.04094 (2019)

30. Warstadt, A., Parrish, A., Liu, H., Mohananey, A., Peng, W., Wang, S.F., Bowman, S.R.: Blimp: The benchmark of linguistic minimal pairs for english. Transactions of the Association for Computational Linguistics **8**, 377–392 (2020)
31. Wu, B., Zhang, Z., Zhao, H.: Graph-free multi-hop reading comprehension: A select-to-guide strategy. arXiv preprint arXiv:2107.11823 (2021)
32. Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W.W., Salakhutdinov, R., Manning, C.D.: Hotpotqa: A dataset for diverse, explainable multi-hop question answering. arXiv preprint arXiv:1809.09600 (2018)