Unsupervised Question Decomposition for Question Answering

Ethan Perez[1 2] **Patrick Lewis**[1 3]
Wen-tau Yih[1] **Kyunghyun Cho**[2 4*] **Douwe Kiela**[1]
[1]Facebook AI Research, [2]New York University,
[3]University College London, [4]CIFAR Azrieli Global Scholar
perez@nyu.edu

Abstract

We aim to improve question answering (QA) by decomposing hard questions into simpler sub-questions that existing QA systems are capable of answering. Since labeling questions with decompositions is cumbersome, we take an unsupervised approach to produce sub-questions, also enabling us to leverage millions of questions from the internet. Specifically, we propose an algorithm for One-to-N Unsupervised Sequence transduction (ONUS) that learns to map one hard, multi-hop question to many simpler, single-hop sub-questions. We answer sub-questions with an off-the-shelf QA model and give the resulting answers to a recomposition model that combines them into a final answer. We show large QA improvements on HotpotQA over a strong baseline on the original, out-of-domain, and multi-hop dev sets. ONUS automatically learns to decompose different kinds of questions, while matching the utility of supervised and heuristic decomposition methods for QA and exceeding those methods in fluency. Qualitatively, we find that using sub-questions is promising for shedding light on why a QA system makes a prediction.[1]

1 Introduction

It has been a long-standing challenge in AI to answer questions of any level of difficulty (Winograd, 1991). Question answering (QA) systems struggle to answer complex questions such as *"What profession do H. L. Mencken and Albert Camus have in common?"* since the required information is scattered in different places (Yang et al., 2018). However, QA systems accurately answer simpler, related questions such as *"What profession does H. L. Mencken have?"* and *"Who was Albert Camus?"* (Petrochuk and Zettlemoyer, 2018). Thus, a promising strategy to answer hard questions is divide-and-conquer: decompose a hard question into simpler sub-questions, answer the sub-questions with a QA system, and recompose the resulting answers into a final answer, as shown in Figure 1. This approach leverages strong performance on simple questions to help answer harder questions (Christiano et al., 2018).

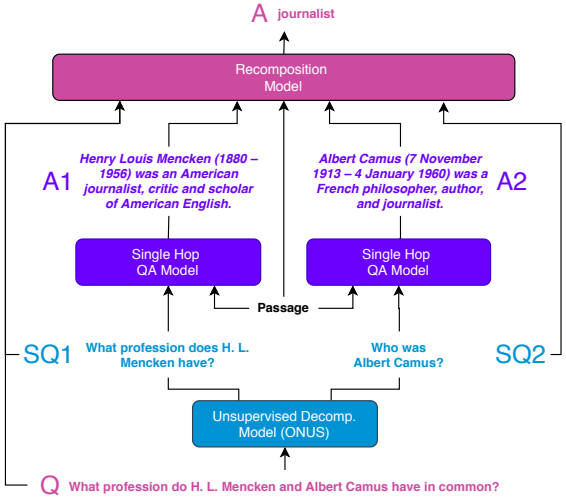

Figure 1: *Overview*: Using unsupervised learning, we decompose a multi-hop question into single-hop sub-questions, whose predicted answers are given to a recomposition model to predict the final answer.

Existing work decomposes questions using a combination of hand-crafted heuristics, rule-based algorithms, and learning from supervised decompositions (Talmor and Berant, 2018; Min et al., 2019b), which each require significant human effort. For example, DecompRC (Min et al., 2019b) decomposes some questions using supervision and other questions using a heuristic algorithm with fine-grained, special case handling based on part-

[*] KC was a part-time research scientist at Facebook AI Research while working on this paper.

[1]Our code, data, and pretrained models are available at https://github.com/facebookresearch/UnsupervisedDecomposition.

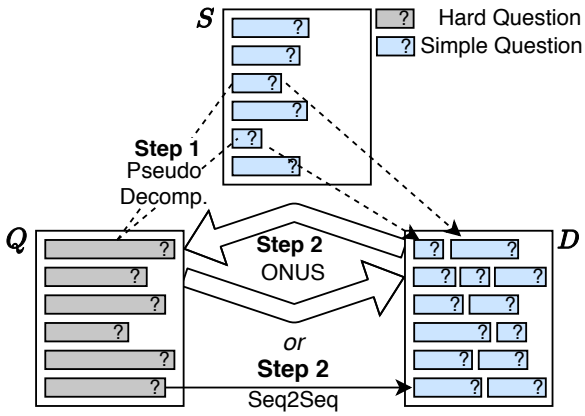

Figure 2: *One-to-N Unsupervised Sequence transduction (ONUS)*: **Step 1**: We create a corpus of pseudo-decompositions D by finding candidate sub-questions from a simple question corpus S which are similar to a multi-hop question in Q. **Step 2**: We learn to map multi-hop questions to decompositions using Q and D as training data, via either standard sequence-to-sequence learning (Seq2Seq) or unsupervised sequence-to-sequence learning (for ONUS).

of-speech tags and over 50 keywords. Prior work also assumes that sub-questions only consist of words from the question, which is not always true. Decomposing arbitrary questions requires sophisticated natural language generation, which often relies on many, high-quality supervised examples. Instead of using supervision, we find it possible to decompose questions in a *fully unsupervised* way.

We propose an algorithm for One-to-N Unsupervised Sequence transduction (ONUS) that learns to map from the distribution of hard questions to that of *many* simple questions. First, we automatically create a noisy "pseudo-decomposition" for each hard question by using embedding similarity to retrieve sub-question candidates. We mine over 10M possible sub-questions from Common Crawl with a classifier, showcasing the effectiveness of parallel corpus mining, a common approach in machine translation (Xu and Koehn, 2017; Artetxe and Schwenk, 2019), for QA. Second, we train a decomposition model on the mined data with unsupervised sequence-to-sequence learning, allowing ONUS to improve over pseudo-decompositions. As a result, we are able to train a large transformer model to generate decompositions, surpassing the fluency of heuristic/extractive decompositions. Figure 2 overviews our approach to decomposition.

We validate ONUS on multi-hop QA, where questions require reasoning over multiple pieces of evidence. We use an off-the-shelf single-hop QA model to answer decomposed sub-questions. Then, we give sub-questions and their answers to a recomposition model to combine into a final answer. We evaluate on three dev sets for HotpotQA, a standard benchmark for multi-hop QA (Yang et al., 2018), including two challenge sets.

ONUS proves to be a powerful tool for QA in the following ways. First, QA models that use decompositions outperform a strong RoBERTa baseline (Liu et al., 2019; Min et al., 2019a) by 3.1 points in F1 on the original dev set, 10 points on the out-of-domain dev set from Min et al. (2019b), and 11 points on the multi-hop dev set from Jiang and Bansal (2019a). Our method is competitive with state-of-the-art methods SAE (Tu et al., 2020) and HGN (Fang et al., 2019) that use additional, strong supervision on which sentences are relevant to the question. Second, our analysis shows that sub-questions improve multi-hop QA by using the single-hop QA model to retrieve question-relevant text. Qualitative examples illustrate how the retrieved text adds a level of interpretability to otherwise black-box, neural QA models. Third, ONUS automatically learns to generate useful decompositions for all four question types in HotpotQA, highlighting the general nature of ONUS over prior work, such as IBM Watson (Ferrucci et al., 2010) and DecompRC (Min et al., 2019b), which decompose different question types separately. Without finetuning, our trained ONUS model can even decompose some questions in visual QA (Johnson et al., 2017b) and knowledge-base QA (Talmor and Berant, 2018), as well as claims in fact verification (Thorne et al., 2018), suggesting promising future avenues in other domains.

2 Method

We now formulate the problem and describe our high-level approach, with further details in §3. The goal of this work is to leverage a QA model that is accurate on simple questions for answering hard questions, without using annotated question decompositions. Here, we consider simple questions to be "single-hop" questions that require reasoning over one paragraph or piece of evidence, and we consider hard questions to be "multi-hop." Our aim is to train a multi-hop QA model M to provide the correct answer a to a multi-hop question q about a given context c (e.g., several paragraphs). Normally, we would train M to maximize $\log p_M(a|c,q)$. To facilitate learn-

ing, we leverage a single-hop QA model that may be queried with sub-questions $s_1, \ldots, s_N$, whose "sub-answers" $a_1, \ldots, a_N$ may be given to M. M may then maximize the potentially easier objective $\log p_M(a|c, q, [s_1, a_1], \ldots, [a_N, s_N])$.

Supervised decomposition models learn to map each question $q \in Q$ to a decomposition $d = [s_1; \ldots; s_N]$ of N sub-questions $s_n \in S$ using annotated (q, d) examples. In this work, we do not assume access to strong (q, d) supervision. To leverage the single-hop QA model without supervision, we follow a three-stage approach: 1) map a question q into sub-questions $s_1, \ldots, s_N$ via unsupervised techniques, 2) find sub-answers $a_1, \ldots, a_N$ with the single-hop QA model, and 3) use $s_1, \ldots, s_N$ and $a_1, \ldots, a_N$ to predict a.

2.1 Unsupervised Question Decomposition

To train an unsupervised decomposition model, we need suitable data. We assume access to a hard question corpus Q and simple question corpus S. Instead of using supervised (q, d) examples, we design an algorithm that creates pseudo-decompositions d' to form (q, d') pairs from Q and S using an unsupervised method (§2.1.1). We then train a model to map q to a decomposition. We explore learning to decompose with standard and unsupervised sequence-to-sequence learning (§2.1.2).

2.1.1 Creating Pseudo-Decompositions

Inspired by Zhou et al. (2015) in question retrieval, we create a pseudo-decomposition set $d' = \{s_1; \ldots; s_N\}$ for each $q \in Q$ by retrieving simple question s_i from S. We concatenate $s_1; \ldots; s_N$ to form d' used downstream. N may potentially vary based on q. To retrieve useful simple questions for answering q, we face a joint optimization problem. We want sub-questions that are both (i) similar to q according to a metric f (first term) and (ii) maximally diverse (second term), so our objective is:

$$\underset{d' \subset S}{\operatorname{argmax}} \sum_{s_i \in d'} f(q, s_i) - \sum_{s_i, s_j \in d', i \neq j} f(s_i, s_j) \quad (1)$$

2.1.2 Learning to Decompose

With the above pseudo-decompositions, we explore various decomposition methods (details in §3.2.3):

PseudoD We use sub-questions from pseudo-decompositions directly in downstream QA.

Sequence-to-Sequence (Seq2Seq) We train a Seq2Seq model p_θ to maximize $\log p_\theta(d'|q)$.

One-to-N Unsupervised Sequence transduction (ONUS) We use unsupervised learning to map one question to N sub-questions. We start with paired (q, d') but do not learn from the pairing because it is noisy. Instead, we use unsupervised Seq2Seq methods to learn a $q \rightarrow d$ mapping.

2.2 Answering Sub-Questions

To answer the generated sub-questions, we use an off-the-shelf QA model. The QA model may answer sub-questions using any free-form text (i.e., a word, phrase, sentence, etc.). Any QA model is suitable, so long as it can accurately answer simple questions in S. We thus leverage good accuracy on questions in S to help answer questions in Q.

2.3 Learning to Recompose

Downstream QA systems may use sub-questions and sub-answers in various ways. We train a recomposition model to combine the decomposed sub-questions/answers into a final answer, when also given the original input (context+question).

3 Experimental Setup

We now detail the implementation of our approach.

3.1 Question Answering Task

We test ONUS on HOTPOTQA, a standard multi-hop QA benchmark. Questions require information from two distinct Wikipedia paragraphs to answer (*"Who is older, Annie Morton or Terry Richardson?"*). For each question, HOTPOTQA provides 10 context paragraphs from Wikipedia. Two paragraphs contain question-relevant sentences called "supporting facts," and the remaining paragraphs are irrelevant, "distractor paragraphs." Answers in HOTPOTQA are either `yes`, `no`, or a text span in an input paragraph. Accuracy is measured with F1 word overlap and Exact Match (EM) between predicted and gold spans.

3.2 Unsupervised Decomposition

3.2.1 Training Data and Question Mining

Supervised decomposition methods are limited by the amount of available human annotation, but our unsupervised method faces no such limitation, similar to unsupervised QA (Lewis et al., 2019). Since we need to train data-hungry Seq2Seq models, we would benefit from large training corpora. A larger simple question corpus will also improve the relevance of retrieved simple questions to the hard

question. Thus, we take inspiration from parallel corpus mining in machine translation (Xu and Koehn, 2017; Artetxe and Schwenk, 2019). We use questions from SQuAD 2 and HotpotQA to form our initial corpora S (single-hop questions) and Q (multi-hop questions), respectively, and we augment Q and S by mining more questions from Common Crawl. First, we select sentences that start with "wh"-words or end in "?" Next, we train an efficient, FastText classifier (Joulin et al., 2017) to classify between questions sampled from Common Crawl, SQuAD 2, and HotpotQA (60K in total). Then, we classify our Common Crawl questions, adding those classified as SQuAD 2 questions to S and those classified as HotpotQA questions to Q. Mining greatly increases the number of single-hop questions (130K $\rightarrow$ 10.1M) and multi-hop questions (90K $\rightarrow$ 2.4M), showing the power of parallel corpus mining in QA. [2]

3.2.2 Creating Pseudo-Decompositions

To create pseudo-decompositions (retrieval-based sub-questions for a given question), we experimented with using a variable number of sub-questions N per question (Appendix §A.1), but we found similar QA results with a fixed $N = 2$, which we use in the remainder for simplicity.

Similarity-based Retrieval To retrieve relevant sub-questions, we embed any text t into a vector $\mathbf{v}_t$ by summing the FastText vectors (Bojanowski et al., 2017)[3] for words in t and use cosine as our similarity metric f.[4] Let q be a multi-hop question with a pseudo-decomposition (s_1^*, s_2^*) and $\hat{\mathbf{v}}$ be the unit vector of $\mathbf{v}$. Since $N = 2$, Eq. 1 simplifies to:

$$(s_1^*, s_2^*) = \underset{\{s_1,s_2\}\in S}{\text{argmax}} \left[\hat{\mathbf{v}}_q^\top \hat{\mathbf{v}}_{s_1} + \hat{\mathbf{v}}_q^\top \hat{\mathbf{v}}_{s_2} - \hat{\mathbf{v}}_{s_1}^\top \hat{\mathbf{v}}_{s_2}\right]$$

The last term requires $O(|S|^2)$ comparisons, which is expensive as $|S| > 10$M. Instead of solving the above equation exactly, we find an approximate pseudo-decomposition (s_1', s_2') by computing over $S' = \text{topK}_{\{s\in S\}} \left[\hat{\mathbf{v}}_q^\top \hat{\mathbf{v}}_s\right]$ with $K = 1000$. We efficiently build S' with FAISS (Johnson et al., 2017a).

Random Retrieval For comparison, we test a random pseudo-decomposition baseline, where we retrieve $s_1, \ldots, s_N$ by sampling uniformly from S.

Editing Pseudo-Decompositions Since sub-questions are retrieval-based, they are often not about the same entities as q. Inspired by retrieve-and-edit methods (e.g., Guu et al., 2018), we replace each sub-question entity not in q with an entity from q of the same type (e.g., "Date" or "Location") if possible.[5] This step is important for PseudoD and Seq2Seq (which would learn to hallucinate entities) but not ONUS (which must reconstruct entities in q from its own decomposition, as discussed next).

3.2.3 Unsupervised Decomposition Models

Pretraining Pretraining is crucial for unsupervised Seq2Seq methods (Artetxe et al., 2018; Lample et al., 2018), so we initialize all decomposition models (Seq2Seq or ONUS) with the same pretrained weights. We warm-start our pretraining with the pretrained, English Masked Language Model (MLM) from Lample and Conneau (2019), a 12-block transformer (Vaswani et al., 2017). We do MLM finetuning for one epoch on Q and pseudo-decompositions D formed via random retrieval, using the final weights to initialize a pretrained encoder-decoder. See Appendix §B.2 for details.

Seq2Seq We finetune the pretrained encoder-decoder using maximum likelihood. We stop training based on validation BLEU between generated decompositions and pseudo-decompositions.

ONUS We finetune the pretrained encoder-decoder with back-translation (Sennrich et al., 2016) and denoising objectives simultaneously, similar to Lample and Conneau (2019) in unsupervised one-to-one translation.[6] For denoising, we produce a noisy input d' by randomly masking, dropping, and locally shuffling tokens in $d \sim D$, and we train a model with parameters θ to maximize $\log p_\theta(d|d')$. We likewise maximize $\log p_\theta(q|q')$ for a noised version q' of $q \sim Q$. For back-translation, we generate a multi-hop question $\hat{q}$ for a decomposition $d \sim D$, and we maximize $\log p_\theta(d|\hat{q})$. Similarly, we maximize $\log p_\theta(q|\hat{d})$ for a model-generated decomposition $\hat{d}$ of $q \sim Q$. To stop training without supervision, we use a modified version of round-trip BLEU (Lample et al., 2018) (see Appendix §B.1 for details). We train on HotpotQA questions Q and their pseudo-decompositions D.[7]

[2] See Appendix §A.3 for details on question classifier.

[3] 300-dim. English Common Crawl vectors: https://fasttext.cc/docs/en/english-vectors.html

[4] We also tried TFIDF and BERT representations but did not see improvements over FastText (see Appendix §A.4).

[5] Entities found with spaCy (Honnibal and Montani, 2017).

[6] www.github.com/facebookresearch/XLM

[7] Using the augmented corpora here did not improve QA.

3.3 Single-hop Question Answering Model

We finetune a pretrained model for single-hop QA following prior work from Min et al. (2019b) on HotpotQA, as described below.[8]

Model Architecture Our model takes in a question and several paragraphs to predict the answer. We compute a separate forward pass on each paragraph (with the question). For each paragraph, the model learns to predict the answer span if the paragraph contains the answer and to predict "no answer" otherwise. We treat `yes` or `no` predictions as spans within the passage (prepended to each paragraph), as in Nie et al. (2019) on HotpotQA. During inference, for the final softmax, we consider all paragraphs as a single chunk. Similar to Clark and Gardner (2018), we subtract a paragraph's "no answer" logit from the logits of all spans in that paragraph, to reduce or increase span probabilities accordingly. In other words, we compute the probability $p(s_p)$ of each span s_p in a paragraph $p \in \{1, \ldots, P\}$ using the predicted span logit $l(s_p)$ and "no answer" paragraph logit $n(p)$ with $p(s_p) \propto e^{l(s_p)-n(p)}$. RoBERTa$_{\text{LARGE}}$ (Liu et al., 2019) is used as our pretrained model.

Training Data and Ensembling Similar to Min et al. (2019b), we train an ensemble of 2 single-hop QA models on SQuAD 2 and the "easy" (single-hop) subset of HotpotQA (see Appendix §C for training details). We average model logits before predicting the answer. We use the single-hop QA ensemble as a black-box model once trained, never training the model on multi-hop questions.

Returned Text Instead of returning only the predicted sub-answer span to the recomposition model, we return the sentence that contains the predicted sub-answer, which is more informative.

3.4 Recomposition Model

Our recomposition model architecture is identical to the single-hop QA model, but the recomposition model also uses sub-questions and sub-answers as input. We append each (sub-question, sub-answer) pair to the question with separator tokens. We train one recomposition model on all of HotpotQA, also including SQuAD 2 examples used to train the single-hop QA model. All reported error margins show the mean and std. dev. across 5 recomposition training runs using the same decompositions.

[8] Code based on `transformers` (Wolf et al., 2019).

Decomp. Method	Pseudo-Decomps.	HotpotQA Dev F1 Orig	Multi	OOD
✗	✗ (1hop)	66.7	63.7	66.5
✗	✗ (Baseline)	$77.0_{\pm.2}$	$65.2_{\pm.2}$	$67.1_{\pm.5}$
PseudoD	Random	$78.4_{\pm.2}$	$70.9_{\pm.2}$	$70.7_{\pm.4}$
	FastText	$78.9_{\pm.2}$	$72.4_{\pm.1}$	$72.0_{\pm.1}$
Seq2Seq	Random	$77.7_{\pm.2}$	$69.4_{\pm.3}$	$70.0_{\pm.7}$
	FastText	$78.9_{\pm.2}$	$73.1_{\pm.2}$	$73.0_{\pm.3}$
ONUS	Random	$79.8_{\pm.1}$	$76.0_{\pm.2}$	$76.5_{\pm.2}$
	FastText	$\mathbf{80.1}_{\pm.2}$	$\mathbf{76.2}_{\pm.1}$	$\mathbf{77.1}_{\pm.1}$
DecompRC*		$79.8_{\pm.2}$	$76.3_{\pm.4}$	$77.7_{\pm.2}$
SAE (Tu et al., 2020) †		80.2	61.1	62.6
HGN (Fang et al., 2019) †		82.2	78.9‡	76.1‡

	Ours	SAE†	HGN†
Test (EM/F1)	66.33/79.34	66.92/79.62	69.22/82.19

Table 1: Unsupervised decompositions significantly improve F1 on HotpotQA over the baseline and single-hop QA model used to answer sub-questions ("1hop"). On all dev sets and the test set, we achieve similar F1 to methods that use supporting fact supervision (†). (*) We test supervised/heuristic decompositions from Min et al. (2019b). (‡) Scores are approximate due to mismatched Wikipedia dumps.

4 Results on Question Answering

We compare variants of our approach that use different learning methods and different pseudo-decomposition training sets. As a baseline, we compare RoBERTa with decompositions to RoBERTa without decompositions. We use the best hyperparameters for the baseline to train our RoBERTa models with decompositions (see Appendix §D.3 for hyperparameters).

We report results on 3 dev set versions: (1) the original version,[9] (2) the multi-hop version from Jiang and Bansal (2019a) who created some distractor paragraphs adversarially to test multi-hop reasoning, and (3) the out-of-domain (OOD) version from Min et al. (2019b) who retrieved distractor paragraphs with the same procedure as the original version but excluded the original paragraphs.

Main Results Table 1 shows how unsupervised decompositions affect QA. Our RoBERTa baseline does quite well on HotpotQA (77.0 F1), in line with Min et al. (2019a) who achieved strong results using a BERT-based version of the model (Devlin et al., 2019). We achieve large gains over the RoBERTa baseline by simply adding sub-questions and sub-answers to the input. Using decompositions from ONUS trained on FastText

[9] Test set is private, so we randomly halve the dev set to form validation/held-out dev sets. Our codebase has our splits.

Q-Type	Using Decomps. ✗	Using Decomps. ✓
Bridge	$80.1_{\pm.2}$	**81.7**$_{\pm.4}$
Comp.	$73.8_{\pm.4}$	**80.1**$_{\pm.3}$
Inters.	$79.4_{\pm.6}$	**82.3**$_{\pm.5}$
1-hop	$73.9_{\pm.6}$	**76.9**$_{\pm.6}$

SQs	SAs	QA F1
✗	✗	$77.0_{\pm.2}$
✓	Sent.	$80.1_{\pm.2}$
✓	Span	$77.8_{\pm.3}$
✓	Rand.	$76.9_{\pm.2}$
✓	✗	$76.9_{\pm.2}$
✗	Sent.	$80.2_{\pm.1}$

Table 2: **Left**: Decompositions improve QA F1 for all 4 HOTPOTQA types. **Right (Ablation)**: QA model F1 when trained with various sub-answers: the sentence of the predicted sub-answer, predicted sub-answer span, or random entity from the context. We also train models with (✓) or without (✗) sub-questions/sub-answers.

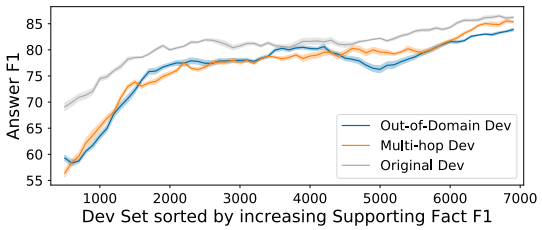

Figure 3: Multi-hop QA is better when the single-hop QA model answers with the ground truth "supporting fact" sentences. We plot mean and std. over 5 QA runs.

pseudo-decompositions, we find a gain of 3.1 F1 on the original dev set, 11 F1 on multi-hop dev, and 10 F1 on OOD dev. ONUS decompositions even match the performance of using supervised and heuristic decompositions from DECOMPRC (i.e., 80.1 vs. 79.8 F1 on the original dev set).

Pseudo-decomposition and ONUS training both contribute to decomposition quality. FastText pseudo-decompositions themselves provide an improvement in QA over the baseline (e.g., 72.0 vs. 67.1 F1 on OOD dev) and over random pseudo-decompositions (70.7 F1), validating our retrieval-based algorithm for creating pseudo-decompositions. Seq2Seq trained on FastText pseudo-decompositions achieves comparable gains to FastText pseudo-decompositions (73.0 F1 on OOD dev), validating the quality of pseudo-decompositions as training data. As hypothesized, ONUS improves over PseudoD and Seq2Seq by learning to align hard questions and pseudo-decompositions while ignoring the noisy pairing (77.1 F1 on OOD dev). ONUS is relatively robust to the training data used but still improves further by using FastText vs. Random pseudo-decompositions (77.1 vs. 76.5 F1 on OOD dev).

We submitted the best QA approach based on dev evaluation (using ONUS trained on FastText pseudo-decompositions) for hidden test evaluation. We achieved a test F1 of 79.34 and Exact Match (EM) of 66.33. Our approach is competitive with state-of-the-art systems SAE (Tu et al., 2020) and HGN (Fang et al., 2019), which both (unlike us) learn from strong, supporting-fact supervision about which sentences are relevant to the question.

4.1 Question Type Breakdown

To understand where decompositions help, we break down QA accuracy across 4 question types from Min et al. (2019b). "Bridge" questions ask about an entity not explicitly mentioned (*"When was Erik Watts' father born?"*). "Intersection" questions ask to find an entity that satisfies multiple separate conditions (*"Who was on CNBC and Fox News?"*). "Comparison" questions ask to compare a property of two entities (*"Which is taller, Momhil Sar or K2?"*). "Single-hop" questions are answerable using single-hop shortcuts or single-paragraph reasoning (*"Where is Electric Six from?"*). We split the original dev set into the 4 types using the supervised type classifier from Min et al. (2019b). Table 2 (left) shows F1 scores for ROBERTA with and without decompositions across the 4 types.

ONUS decompositions improve QA across all types. Our single decomposition model does not need to be tailored to the question type, unlike Min et al. (2019b) who use a different model per question type. For single-hop questions, our QA approach does not require falling back to a single-hop QA model and instead learns to leverage decompositions in that case also (76.9 vs. 73.9 F1).

4.2 Answers to Sub-Questions are Crucial

To measure the usefulness of sub-questions and sub-answers, we train the recomposition model with various, ablated inputs, as shown in Table 2 (right). Sub-answers are crucial to improving QA, as sub-questions with no answers or random answers do not help (76.9 vs. 77.0 F1 for the baseline). Only when sub-answers are provided do we see improved QA, with or without sub-questions (80.1 and 80.2 F1, respectively). It is important to provide the sentence containing the predicted answer span instead of the answer span alone (80.1 vs. 77.8 F1, respectively), though the answer span alone still improves over the baseline (77.0 F1).

4.3 How Do Decompositions Help?

Decompositions help by retrieving important supporting evidence to answer questions. Fig. 3 shows

Q1: Who is older, Annie Morton or Terry Richardson?
SQ$_1$: Who is Annie Morton?
⌞ Annie Morton (born October 8, 1970) is an American model born in Pennsylvania.
SQ$_2$: When was Terry Richardson born?
⌞ Kenton Terry Richardson (born 26 July 1999) is an English professional footballer who plays as a defender for League Two side Hartlepool United.
Â: Annie Morton

Q2: How many copies of Roald Dahl's variation on a popular anecdote sold?
SQ$_1$: How many copies of Roald Dahl's?
⌞ His books have sold more than 250 million copies worldwide.
SQ$_2$ What is the name of the variation on a popular anecdote?
⌞ "Mrs. Bixby and the Colonel's Coat" is a short story by Roald Dahl that first appeared in the 1959 issue of Nugget.
Â: more than 250 million

Q3: Are both Coldplay and Pierre Bouvier from the same country?
SQ$_1$: Where are Coldplay and Coldplay from?
⌞ Coldplay are a British rock band formed in 1996 by lead vocalist and keyboardist Chris Martin and lead guitarist Jonny Buckland at University College London (UCL).
SQ$_2$: What country is Pierre Bouvier from?
⌞ Pierre Charles Bouvier (born 9 May 1979) is a Canadian singer, songwriter, musician, composer and actor who is best known as the lead singer and guitarist of the rock band Simple Plan.
Â: No

Table 3: Example sub-questions generated by our model, along with predicted sub-answer sentences (answer span underlined) and final predicted answer.

that QA improves when the sub-answer sentences are gold "supporting facts." We retrieve these without relying on strong, supporting fact supervision, unlike many state-of-the-art models (Tu et al., 2020; Fang et al., 2019; Nie et al., 2019).[10]

4.4 Example Decompositions

To illustrate how decompositions help, Table 3 shows example sub-questions from ONUS with predicted sub-answers. Sub-questions are single-hop questions relevant to the multi-hop question. The single-hop QA model returns relevant sub-answers, sometimes despite under-specified (**Q2, SQ$_1$**) or otherwise imperfect sub-questions (**Q3, SQ$_1$**). The recomposition model returns an answer consistent with the sub-answers. Furthermore, the sub-answers used for QA are in natural language, adding a level of interpretability to otherwise black-box, neural QA models. Decompositions are largely extractive, copying from the multi-hop question rather than hallucinating new entities, which helps generate relevant sub-questions. Appendix Table 7 shows decompositions from our trained ONUS model, without further finetuning, on image-based questions (CLEVR; Johnson et al., 2017b), knowledge-base questions (ComplexWebQuestions; Talmor and Berant, 2018), and even claims in fact verification (FEVER; Thorne et al., 2018), which suggests promising future avenues for our approach in other domains and highlights the general nature of the proposed method.

[10] See Appendix §B.3 for supporting fact scores.

Decomp. Method	GPT2 NLL	% Well-Formed	Edit Dist.	Length Ratio
ONUS	5.56	60.9	5.96	1.08
DecompRC	6.04	32.6	7.08	1.22

Table 4: Analysis of sub-questions produced by our method vs. the supervised+heuristic method of Min et al. (2019b). Left-to-right: Negative Log-Likelihood according to GPT2 (lower is better), % classified as Well-Formed, Edit Distance between decomposition and multi-hop question, and token-wise Length Ratio between decomposition and multi-hop question.

5 Analysis

To better understand our system, we now analyze our pipeline by examining the model for each stage: decomposition, single-hop QA, and recomposition.

5.1 Unsupervised Decomposition Model

Intrinsic Evaluation of Decompositions We evaluate the quality of decompositions on other metrics aside from downstream QA. To measure the fluency of decompositions, we compute the likelihood of decompositions using the pretrained GPT-2 language model (Radford et al., 2019). We train a BERT$_{\text{BASE}}$ classifier on the question-wellformedness dataset of Faruqui and Das (2018), and we use the classifier to estimate the proportion of sub-questions that are well-formed. We measure how abstractive decompositions are by computing (i) the token Levenstein distance between the multi-hop question and its generated decomposition and (ii) the ratio between the length of the decomposition and the length of the multi-hop question. We compare ONUS to DecompRC (Min et al., 2019b), a supervised+heuristic decomposition method.

As shown in Table 4, ONUS decompositions are more natural and well-formed than DecompRC decompositions. As an example, for Table 3 **Q3**, DecompRC produces the sub-questions "Is Coldplay from which country?" and "Is Pierre Bouvier

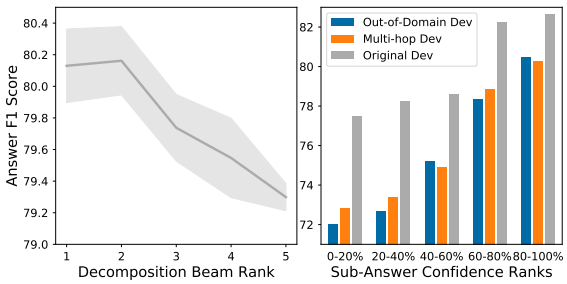

Figure 4: **Left**: We decode decompositions with beam search and use n[th]-ranked hypothesis as a question decomposition. We plot the F1 of a recomposition model trained to use the n[th]-ranked decomposition. **Right**: Multi-hop QA is better when the single-hop QA model places high probability on its sub-answer.

from which country?" ONUS decompositions are also closer in edit distance and length to the multi-hop question, consistent with our observation that our decomposition model is largely extractive.

Quality of Decomposition Model A well-trained decomposition model should place higher probability on decompositions that are more helpful for QA. We generate $N = 5$ hypotheses from our best decomposition model using beam search, and we train a recomposition model to use the n^{th}-ranked hypothesis as a question decomposition (Figure 4, left). QA accuracy decreases as we use lower probability decompositions, but accuracy remains relatively robust, at most decreasing from 80.1 to 79.3 F1. The limited drop suggests that decompositions are still useful if they are among the model's top hypotheses, another indication that ONUS is trained well for decomposition.

5.2 Single-hop Question Answering Model

Sub-Answer Confidence Figure 4 (right) shows that the single-hop model's sub-answer confidence correlates with downstream multi-hop QA accuracy on all dev sets. A low confidence sub-answer may be indicative of (i) an unanswerable or ill-formed sub-question or (ii) a sub-answer that is more likely to be incorrect. In both cases, the single-hop QA model is less likely to retrieve useful supporting evidence for answering the multi-hop question.

Changing the Single-hop QA Model We find that our approach is robust to the single-hop QA model used. We test the $\text{BERT}_{\text{BASE}}$ ensemble from Min et al. (2019b) as the single-hop QA model. The model performs much worse compared to our $\text{RoBERTA}_{\text{LARGE}}$ single-hop ensemble on HotpotQA itself (56.3 vs. 66.7 F1). However, the model results in similar QA when used to answer single-hop sub-questions within our larger system (79.9 vs. 80.1 F1 for our ensemble).

Recomposition Model	QA F1 (w/o → w/ Decomps.)
$\text{BERT}_{\text{BASE}}$	71.8±.4 → 73.0±.4
$\text{BERT}_{\text{LARGE}}$	76.4±.2 → 79.0±.1
$\text{RoBERTA}_{\text{LARGE}}$	77.0±.3 → 80.1±.2

Table 5: Better models gain more from decomposition.

5.3 Recomposition Model

Varying the Base Model To understand how decompositions impact performance as the recomposition model gets stronger, we vary the base pre-trained model. Table 5 shows the impact of adding decompositions to $\text{BERT}_{\text{BASE}}$, $\text{BERT}_{\text{LARGE}}$, and finally $\text{RoBERTA}_{\text{LARGE}}$ (see Appendix §D.3 for hyperparameters). The gain from using decompositions grows with strength of the recomposition model. Decompositions improve QA by 1.2 F1 for a $\text{BERT}_{\text{BASE}}$ model, by 2.6 F1 for the stronger $\text{BERT}_{\text{LARGE}}$ model, and by 3.1 F1 for our best $\text{RoBERTA}_{\text{LARGE}}$ model.

6 Related Work

Answering complex questions has been a long-standing challenge in natural language processing. Prior work explored decomposing questions with supervision and heuristic algorithms. IBM Watson (Ferrucci et al., 2010) decomposes questions into sub-questions in multiple ways or not at all. DecompRC (Min et al., 2019b) largely frames sub-questions as extractive spans of a question, learning to predict span-based sub-questions via supervised learning on human annotations. In other cases, DecompRC decomposes a multi-hop question using a heuristic algorithm or not at all. Watson and DecompRC use special case handling to decompose different questions, while our algorithm is fully automated and requires little hand-engineering.

More traditional, semantic parsing methods map questions to compositional programs, whose sub-programs can be viewed as question decompositions in a formal language (Talmor and Berant, 2018; Wolfson et al., 2020). Examples include classical QA systems like SHRDLU (Winograd, 1972) and LUNAR (Woods et al., 1974), as well as neural Seq2Seq semantic parsers (Dong and Lapata, 2016) and neural module networks (Andreas et al., 2015, 2016). Such methods usually require

strong, program-level supervision to generate programs, as in visual QA (Johnson et al., 2017c) and on HOTPOTQA (Jiang and Bansal, 2019b). Some models use other forms of strong supervision, e.g., the sentences needed to answer a question, as annotated by HOTPOTQA. Such an approach is taken by SAE (Tu et al., 2020) and HGN (Fang et al., 2019), whose methods may be combined with ours.

Unsupervised decomposition complements strongly and weakly supervised decomposition approaches. Our unsupervised approach enables methods to leverage millions of otherwise unusable questions, similar to work on unsupervised QA (Lewis et al., 2019). When decomposition examples exist, supervised and unsupervised learning can be used in tandem to learn from both labeled and unlabeled examples. Such semi-supervised methods outperform supervised learning for tasks like machine translation (Sennrich et al., 2016). Other work on weakly supervised question generation uses a downstream QA model's accuracy as a signal for learning to generate useful questions. Weakly supervised question generation often uses reinforcement learning (Nogueira and Cho, 2017; Wang and Lake, 2019; Strub et al., 2017; Das et al., 2017; Liang et al., 2018), where an unsupervised initialization can greatly mitigate the issues of exploring from scratch (Jaderberg et al., 2017).

7 Conclusion

We proposed a QA system that answers a question via decomposition, without supervised question decompositions, using three stages: (1) decompose a question into many sub-questions using One-to-N Unsupervised Sequence transduction (ONUS), (2) answer sub-questions with an off-the-shelf QA system, and (3) recompose sub-answers into a final answer. When evaluated on three HOTPOTQA dev sets, our approach significantly improved QA over an equivalent model that did not use decompositions. Our approach relies only on the final answer as supervision but works as effectively as state-of-the-art methods that rely on much stronger supervision, such as supporting fact labels or example decompositions. We found that ONUS generates fluent sub-questions whose answers often match the gold-annotated, question-relevant text. Overall, this work opens up exciting avenues for leveraging methods in unsupervised learning and natural language generation to improve the interpretability and generalization of machine learning systems.

Acknowledgments

EP's work at NYU is supported by the NSF Graduate Research Fellowship and the Open Philanthropy AI Fellowship. KC's work at NYU is partly supported by Samsung Advanced Institute of Technology (Next Generation Deep Learning: from pattern recognition to AI) and Samsung Research (Improving Deep Learning using Latent Structure). KC also thanks Naver, eBay, NVIDIA, and NSF Award 1922658 for support. HOTPOTQA and SQUAD are licensed under CC BY-SA 4.0. FEVER 1.0 and 2.0 are licensed under CC BY-SA 3.0. We thank Paul Christiano, Sebastian Riedel, He He, Jonathan Berant, Alexis Conneau, Jiatao Gu, Sewon Min, Yixin Nie, Lajanugen Logeswaran, Adam Fisch, Elman Mansimov, Iacer Calixto, Richard Pang, and our anonymous reviewers for helpful feedback, as well as Yichen Jiang and Peng Qi for help with evaluation.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2015. Neural module networks. *CVPR*, pages 39–48.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016. Learning to compose neural networks for question answering. In *NAACL*, pages 1545–1554, San Diego, California. Association for Computational Linguistics.

Mikel Artetxe, Gorka Labaka, Eneko Agirre, and Kyunghyun Cho. 2018. Unsupervised neural machine translation. In *ICLR*.

Mikel Artetxe and Holger Schwenk. 2019. Margin-based parallel corpus mining with multilingual sentence embeddings. In *ACL*, pages 3197–3203, Florence, Italy. Association for Computational Linguistics.

Piotr Bojanowski, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2017. Enriching word vectors with subword information. *TACL*, 5:135–146.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *ACL*, pages 1870–1879, Vancouver, Canada. Association for Computational Linguistics.

Paul Francis Christiano, Buck Shlegeris, and Dario Amodei. 2018. Supervising strong learners by amplifying weak experts. *CoRR*, abs/1810.08575.

Christopher Clark and Matt Gardner. 2018. Simple and effective multi-paragraph reading comprehension. In *ACL*, pages 845–855, Melbourne, Australia. Association for Computational Linguistics.

Abhishek Das, Satwik Kottur, José M.F. Moura, Stefan Lee, and Dhruv Batra. 2017. Learning cooperative visual dialog agents with deep reinforcement learning. In *ICCV*.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. BERT: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*, pages 4171–4186.

Li Dong and Mirella Lapata. 2016. Language to logical form with neural attention. In *ACL*, pages 33–43, Berlin, Germany. Association for Computational Linguistics.

Yuwei Fang, Siqi Sun, Zhe Gan, Rohit Pillai, Shuohang Wang, and Jingjing Liu. 2019. Hierarchical graph network for multi-hop question answering.

Manaal Faruqui and Dipanjan Das. 2018. Identifying well-formed natural language questions. In *EMNLP*, pages 798–803, Brussels, Belgium. Association for Computational Linguistics.

David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A. Kalyanpur, Adam Lally, J. William Murdock, Eric Nyberg, John Prager, Nico Schlaefer, and Chris Welty. 2010. Building watson: An overview of the deepqa project. *AI Magazine*, 31(3):59–79.

Kelvin Guu, Tatsunori B. Hashimoto, Yonatan Oren, and Percy Liang. 2018. Generating sentences by editing prototypes. *TACL*, 6:437–450.

Matthew Honnibal and Ines Montani. 2017. spaCy 2: Natural language understanding with Bloom embeddings, convolutional neural networks and incremental parsing. To appear.

Max Jaderberg, Volodymyr Mnih, Wojciech Marian Czarnecki, Tom Schaul, Joel Z Leibo, David Silver, and Koray Kavukcuoglu. 2017. Reinforcement learning with unsupervised auxiliary tasks. In *ICLR*.

Yichen Jiang and Mohit Bansal. 2019a. Avoiding reasoning shortcuts: Adversarial evaluation, training, and model development for multi-hop QA. In *ACL*, pages 2726–2736, Florence, Italy. Association for Computational Linguistics.

Yichen Jiang and Mohit Bansal. 2019b. Self-assembling modular networks for interpretable multi-hop reasoning. In *EMNLP*, Hong Kong, China. Association for Computational Linguistics.

Jeff Johnson, Matthijs Douze, and Hervé Jégou. 2017a. Billion-scale similarity search with gpus. *CoRR*, abs/1702.08734.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Li Fei-Fei, C. Lawrence Zitnick, and Ross Girshick. 2017b. CLEVR: A diagnostic dataset for compositional language and elementary visual reasoning. In *CVPR*.

Justin Johnson, Bharath Hariharan, Laurens van der Maaten, Judy Hoffman, Li Fei-Fei, C Lawrence Zitnick, and Ross Girshick. 2017c. Inferring and executing programs for visual reasoning. In *ICCV*.

Armand Joulin, Edouard Grave, Piotr Bojanowski, and Tomas Mikolov. 2017. Bag of tricks for efficient text classification. In *EACL*, pages 427–431, Valencia, Spain. Association for Computational Linguistics.

Guillaume Lample and Alexis Conneau. 2019. Cross-lingual language model pretraining. In *NeurIPS*.

Guillaume Lample, Alexis Conneau, Ludovic Denoyer, and Marc'Aurelio Ranzato. 2018. Unsupervised machine translation using monolingual corpora only. In *ICLR*.

Patrick Lewis, Ludovic Denoyer, and Sebastian Riedel. 2019. Unsupervised question answering by cloze translation. In *ACL*, pages 4896–4910, Florence, Italy. Association for Computational Linguistics.

Chen Liang, Mohammad Norouzi, Jonathan Berant, Quoc V Le, and Ni Lao. 2018. Memory augmented policy optimization for program synthesis and semantic parsing. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *NeurIPS*, pages 9994–10006. Curran Associates, Inc.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. RoBERTa: A robustly optimized bert pretraining approach. *CoRR*, abs/1907.11692.

Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. 2018. Mixed precision training. In *ICLR*.

Sewon Min, Eric Wallace, Sameer Singh, Matt Gardner, Hannaneh Hajishirzi, and Luke Zettlemoyer. 2019a. Compositional questions do not necessitate multi-hop reasoning. In *ACL*, pages 4249–4257, Florence, Italy. Association for Computational Linguistics.

Sewon Min, Victor Zhong, Luke Zettlemoyer, and Hannaneh Hajishirzi. 2019b. Multi-hop reading comprehension through question decomposition and rescoring. In *ACL*, pages 6097–6109, Florence, Italy. Association for Computational Linguistics.

Yixin Nie, Songhe Wang, and Mohit Bansal. 2019. Revealing the importance of semantic retrieval for machine reading at scale. In *EMNLP*.

Rodrigo Nogueira and Kyunghyun Cho. 2017. Task-oriented query reformulation with reinforcement learning. In *EMNLP*, pages 574–583, Copenhagen, Denmark. Association for Computational Linguistics.

Michael Petrochuk and Luke Zettlemoyer. 2018. SimpleQuestions nearly solved: A new upperbound and baseline approach. In *EMNLP*, pages 554–558, Brussels, Belgium. Association for Computational Linguistics.

Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. 2019. Language models are unsupervised multitask learners.

Rico Sennrich, Barry Haddow, and Alexandra Birch. 2016. Improving neural machine translation models with monolingual data. In *ACL*, pages 86–96, Berlin, Germany. Association for Computational Linguistics.

Florian Strub, Harm de Vries, Jérémie Mary, Bilal Piot, Aaron Courville, and Olivier Pietquin. 2017. End-to-end optimization of goal-driven and visually grounded dialogue systems. In *IJCAI*, pages 2765–2771.

Alon Talmor and Jonathan Berant. 2018. The web as a knowledge-base for answering complex questions. In *NAACL*, pages 641–651, New Orleans, Louisiana. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and verification. *CoRR*, abs/1803.05355.

James Thorne, Andreas Vlachos, Oana Cocarascu, Christos Christodoulopoulos, and Arpit Mittal. 2019. The FEVER2.0 shared task. In *Proceedings of the Second Workshop on Fact Extraction and VERification (FEVER)*, pages 1–6, Hong Kong, China. Association for Computational Linguistics.

Ming Tu, Kevin Huang, Guangtao Wang, Jing Huang, Xiaodong He, and Bowen Zhou. 2020. Select, answer and explain: Interpretable multi-hop reading comprehension over multiple documents. In *AAAI*.

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, Ł ukasz Kaiser, and Illia Polosukhin. 2017. Attention is all you need. In I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *NeurIPS*, pages 5998–6008. Curran Associates, Inc.

Ziyun Wang and Brenden M. Lake. 2019. Modeling question asking using neural program generation. *CoRR*, abs/1907.09899.

Terry Winograd. 1972. *Understanding Natural Language*. Academic Press, Inc., USA.

Terry Winograd. 1991. *Thinking Machines: Can There Be? Are We?* University of California Press, Berkeley.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, R'emi Louf, Morgan Funtowicz, and Jamie Brew. 2019. Huggingface's transformers: State-of-the-art natural language processing. *CoRR*, abs/1910.03771.

Tomer Wolfson, Mor Geva, Ankit Gupta, Matt Gardner, Yoav Goldberg, Daniel Deutch, and Jonathan Berant. 2020. Break it down: A question understanding benchmark. *TACL*.

W. Woods, R. Kaplan, and B. Nash-Webber. 1974. The lunar sciences natural language information system. Final Report 2378, Bolt, Beranek and Newman, Inc., Cambridge, MA.

Hainan Xu and Philipp Koehn. 2017. Zipporah: a fast and scalable data cleaning system for noisy web-crawled parallel corpora. In *EMNLP*, pages 2945–2950, Copenhagen, Denmark. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *EMNLP*, pages 2369–2380, Brussels, Belgium. Association for Computational Linguistics.

Guangyou Zhou, Tingting He, Jun Zhao, and Po Hu. 2015. Learning continuous word embedding with metadata for question retrieval in community question answering. In *ACL*, pages 250–259, Beijing, China. Association for Computational Linguistics.

Yukun Zhu, Ryan Kiros, Rich Zemel, Ruslan Salakhutdinov, Raquel Urtasun, Antonio Torralba, and Sanja Fidler. 2015. Aligning books and movies: Towards story-like visual explanations by watching movies and reading books. In *ICCV*, page 19–27, USA. IEEE Computer Society.

A Pseudo-Decompositions

Tables 8-10 show examples of pseudo-decompositions and learned decompositions from various models.

A.1 Variable-Length Pseudo-Decompositions

A general algorithm for creating pseudo-decompositions should find a suitable number of sub-questions N for each question. To this end, we compare the objective in Eq. 1 for creating pseudo-decompositions with an alternate objective based on Euclidean distance. This alternate objective has the advantage that the regularization term that encourages sub-question diversity grows more slowly N, disencouraging larger N less:

$$d'^{*} = \underset{d' \subset S}{\operatorname{argmin}} \left|\left| \mathbf{v}_q - \sum_{s \in d'} \mathbf{v}_s \right|\right|_2 \quad (2)$$

We create pseudo-decompositions in an similar way as with Eq. 1, first finding a set of candidate sub-questions $S' \subset S$ with high cosine similarity to $\mathbf{v}_q$. Then, we perform beam search to sequentially choose sub-questions up to a maximum of N sub-questions.

We test pseudo-decomposition objectives by creating synthetic, compositional questions by combining 2-3 single-hop questions with "and." Then, we measure rank of the correct decomposition (a concatenation of the single-hop questions), according to each objective. For $N = 2$, both objectives perform well. For $N = 3$, Eq. 2 achieves a mean reciprocal rank of 30%, while Eq. 1 gets ∼0%. In practice, few questions appear to require $N > 2$ on HOTPOTQA, as we find similar QA accuracy with Eq. 1 (which consistently uses $N = 2$ sub-questions) and Eq. 2 (which mostly uses $N = 2$ but sometimes uses $N = 3$). For example, with Eq. 1 vs. Eq. 2, we find 79.9 vs. 79.4 dev F1 when using the $\text{BERT}_{\text{BASE}}$ ensemble from Min et al. (2019b) to answer sub-questions. Thus, we use Eq. 1 in our main experiments, as it is simpler and faster to compute. Table 8 contains an example where the variable-length decomposition method discussed above (Eq. 2) generates three sub-questions while other methods produce two.

A.2 Impact of Question Corpus Size

In addition to our previous results on FastText vs. Random pseudo-decompositions, we found it important to use a large question corpus to create pseudo-decompositions. QA F1 increased from 79.2 to 80.1 when we trained decomposition models on pseudo-decompositions comprised of questions retrieved from Common Crawl (>10M questions) rather than only SQUAD 2 (∼130K questions), using an appropriately larger beam size for pseudo-decomposition (100 → 1000).

Decomp. Method	Pseudo-Decomps.	HOTPOTQA F1 Dev	Advers.	OOD
✗	✗ (1hop)	66.7	63.7	66.5
✗	✗ (Baseline)	77.0±.2	65.2±.2	67.1±.5
PseudoD	Random	78.4±.2	70.9±.2	70.7±.4
	BERT	78.9±.4	71.5±.3	71.5±.2
	TFIDF	79.2±.3	72.2±.3	72.0±.5
	FastText	78.9±.2	72.4±.1	72.0±.1
Seq2Seq	Random	77.7±.2	69.4±.3	70.0±.7
	BERT	79.1±.3	72.6±.3	73.1±.3
	TFIDF	79.2±.1	73.0±.3	72.9±.3
	FastText	78.9±.2	73.1±.2	73.0±.3
CONUS	Random	79.4±.2	75.1±.2	75.2±.4
	BERT	78.9±.2	74.9±.1	75.2±.2
	TFIDF	78.6±.3	72.4±.4	72.8±.2
	FastText	79.9±.2	76.0±.1	76.9±.1
ONUS	Random	79.8±.1	76.0±.2	76.5±.2
	BERT	79.8±.3	**76.2**±.3	76.7±.3
	TFIDF	79.6±.2	75.5±.2	76.0±.2
	FastText	**80.1**±.2	**76.2**±.1	**77.1**±.1
DecompRC		79.8±.2	76.3±.4	77.7±.2
SAE (Tu et al., 2020)		80.2	61.1	62.6
HGN (Fang et al., 2019)		82.2	78.9	76.1

Table 6: QA F1 scores for all combinations of learning methods and pseudo-decomposition retrieval methods that we tried.

A.3 Question Mining Details

We train a 4-way FastText, bag-of-words classifier to classifier between (1) HOTPOTQA "Bridge"/"Intersection" questions (See §4.1 for definitions), (2) HOTPOTQA "Comparison" questions (See §4.1 for definition), (3) SQuAD 2.0 questions, (4) and Common Crawl questions. We randomly sample 15K examples from each of the above four groups of questions to form our training data. The trained classifier performs well, achieving 95.5% accuracy for HOTPOTQA vs. SQuAD question classification on held-out questions. Questions in Common Crawl that were classified as from HOTPOTQA by the classifier often had more words, conjunctions ("or," "and"), and comparison words ("older," "earlier"), and were generally complex questions.

A.4 Pseudo-Decomposition Retrieval Method

Table 6 shows QA results with pseudo-decompositions retrieved using sum-bag-of-

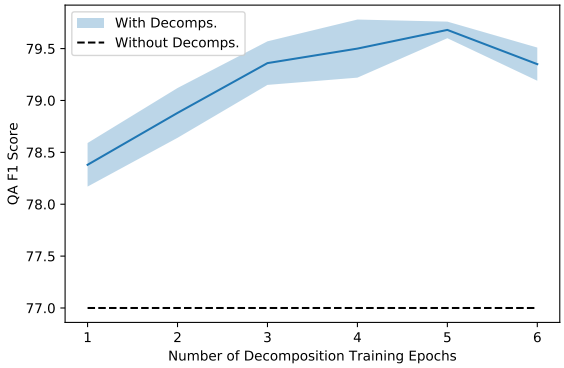

Figure 5: How multi-hop QA accuracy varies over the course of decomposition model training, for one training run of ONUS on FastText pseudo-decompositions. Our unsupervised stopping criterion selects the epoch 3 checkpoint, which performs roughly as well as the best checkpoint (epoch 5).

word representations from FastText, TFIDF, $\text{BERT}_{\text{LARGE}}$ first layer hidden states. We also vary the learning method and include results Curriculum ONUS (CONUS), where we initialize the ONUS approach with the Seq2Seq model trained on the same data.

B Unsupervised Decomposition Model

B.1 Training Procedure

Unsupervised Stopping Criterion To stop ONUS training, we use an unsupervised stopping criterion to avoid relying on a supervised validation set of decompositions. We generate a decomposition $\hat{d}$ for a multi-hop question q, and we measure BLEU between q and the model-generated question $\hat{q}$ for $\hat{d}$, similar to round-trip BLEU in unsupervised one-to-one translation (Lample et al., 2018). We scale round-trip BLEU score by the fraction of "good" decompositions, where a good decomposition has (1) two sub-questions (question marks), (2) no sub-question which contains all words in the multi-hop question, and (3) no sub-question longer than the multi-hop question. We chose these criteria to detect a failure mode; without scaling, decomposition models can achieve perfect round-trip BLEU by copying the multi-hop question as the decomposition. We measure scaled BLEU across multi-hop questions in HotpotQA dev, and we stop training when the metric does not increase for 3 consecutive epochs.

It is possible to stop training the decomposition model based on downstream QA accuracy. However, training a QA model on each decomposition model checkpoint (1) is computationally expensive and (2) ties decompositions to a specific, downstream QA model. In Figure 5, we show downstream QA results across various ONUS checkpoints when using the $\text{BERT}_{\text{BASE}}$ single-hop QA ensemble from Min et al. (2019b). The unsupervised stopping criterion does not significantly hurt downstream QA compared to using a weakly-supervised stopping criterion based on multi-hop QA accuracy.

B.2 Training Hyperparameters

MLM Pretraining We warm-start our pretraining with the 340M parameter, pretrained, English Masked Language Model (MLM) from Lample and Conneau (2019), a 12-block *encoder-only* transformer (Vaswani et al., 2017) trained on Toronto Books Corpus (Zhu et al., 2015) and Wikipedia. We pretrain our encoder for 26 hours (one full epoch on Q) with 8 DGX-1 machines, each with 8, 32GB NVIDIA V100 GPUs interconnected by Infiniband. We use the largest possible batch size (1536), and we choose the best learning rate (3×10^{-5}) based on training loss after a small number of iterations. We chose a maximum sequence length of 128. Other hyperparameters are identical to those from Lample and Conneau (2019) used in unsupervised one-to-one translation. To initialize a pretrained *encoder-decoder* from the encoder-only MLM, we initialize a 6-block encoder with the first 6 MLM blocks, and we initialize a 6-block decoder with the last 6 MLM blocks, randomly initializing the remaining weights as in Lample and Conneau (2019).

ONUS We train each decomposition model with distributed training over 8, 32GB NVIDIA V100 GPUs, lasting roughly 8 hours. We chose the largest batch size that fit in GPU memory (256) and then the largest learning rate that resulted in stable learning early in training (3×10^{-5}). Other hyperparameters are the same as Lample and Conneau (2019).

Seq2Seq We again train each decomposition model with distributed training over 8, 32GB NVIDIA V100 GPUs, lasting roughly 8 hours. We use a large batch size (1024) and chose the largest learning rate which resulted in stable training across the various pseudo-decomposition training corpora from Appendix §A.4 (1×10^{-4}). We keep other training settings and hyperparameters the same as for ONUS.

B.3 Unsupervised Fact Retrieval

Our unsupervised supporting fact retrieval (described in §4.3) achieves 15.7 EM and 55.2 F1 for retrieving the gold supporting facts (sentences) needed to answer HOTPOTQA questions. To our knowledge, there is no prior work on unsupervised fact retrieval on HOTPOTQA to compare against, but our performance approaches early, supervised fact-retrieval methods on HOTPOTQA from Yang et al. (2018) which achieve 59.0 F1.

B.4 Decomposing Questions in Other Tasks

As shown in Table 7, we decompose queries from several other datasets, using our decomposition model trained on only questions in HOTPOTQAand Common Crawl. In particular, we generate subquestions for (1) questions in ComplexWebQuestions (Talmor and Berant, 2018), which are multi-hop questions about knowledge-bases, (2) questions in CLEVR (Johnson et al., 2017b), which are multi-hop questions about images, and (3) claims (statements) in fact-verification challenges, FEVER 1.0 (Thorne et al., 2018) and 2.0 (Thorne et al., 2019). These queries differ significantly from questions in HOTPOTQA in topic, syntactic structure, and/or modality being asked about. Despite such differences, our trained ONUS model often (though not always) generates reasonable subquestions without any further finetuning, providing further evidence of the general nature of our approach and potential for applicability to other domains.

C Single-hop QA Model

To train the single-hop QA model, we largely follow Min et al. (2019b) as described below. We use an ensemble of two models trained on SQUAD 2 and examples from HOTPOTQA labeled as "easy" (single-hop). SQUAD is a single-paragraph QA task, so we adapt it to the multi-paragraph setting by retrieving and appending distractor paragraphs from Wikipedia for each question. We use the TFIDF retriever from DrQA (Chen et al., 2017) to retrieve two distractor paragraphs, which we add to the input for one model in the ensemble. We drop words from the question with a 5% probability to help the model handle any ill-formed subquestions.

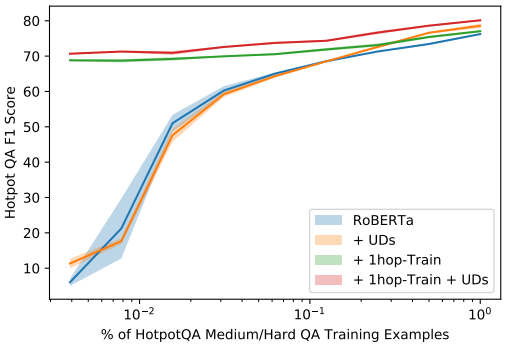

Figure 6: QA F1 of the downstream, recomposition model, with and without unsupervised decompositions (UDs), when varying the amount of training data. We also assess the impact of removing single-hop training data (SQUAD 2.0 and HOTPOTQA"easy" questions).

D Recomposition Model

D.1 Varying Training Set Size

To understand how decompositions impact performance given different amounts of QA training data, we vary the number of multi-hop training examples. We use the "medium" and "hard" level labels in HOTPOTQA to determine which examples are multi-hop. We consider training setups where the recomposition model does or does not use data augmentation via training on hotpot "easy"/single-hop questions and SQUAD 2 questions. Fig. 6 shows the results. Decompositions improve QA, so long as the recomposition model has enough training data to achieve a minimum level of performance (here, roughly 68 F1).

D.2 Improvements across Question Types

To better understand where decompositions improve QA, we examined the improvement over the baseline across various fine-grained splits of our three evaluation sets. Decompositions were roughly as helpful for `yes/no` questions as for questions with a span-based answer. Across our evaluation sets, we did not find a consistent pattern regarding what questions, stratified by "wh-" question-starting words, benefited the most from decompositions. Intuitively, we found larger QA improvements when using decompositions when a sub-answer sentence contained a gold, final answer, as shown in Figure 7.

D.3 Training Hyperparameters

To train ROBERTA$_{\text{LARGE}}$, we fix the number of training epochs to 2, as training longer did not help.

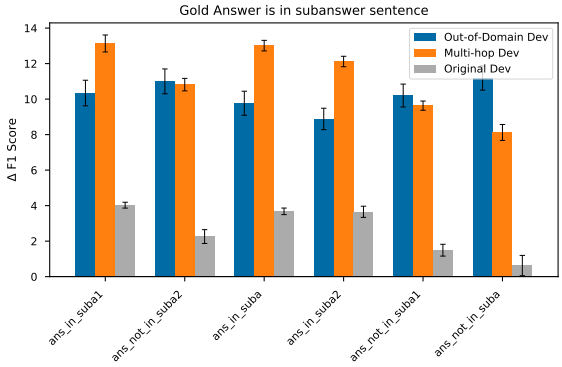

Figure 7: Performance difference between a QA model that does vs. does not use ONUS decompositions, stratified by whether the gold final answer is in a subanswer sentence. We find a larger gain when the subanswer sentence contains the gold, final answer.

We sweep over batch size $\in \{64, 128\}$, learning rate $\in \{1\times10^{-5}, 1.5\times10^{-5}, 2\times10^{-5}, 3\times10^{-5}\}$, and weight decay $\in \{0, 0.1, 0.01, 0.001\}$, similar to the ranges used in the original paper (Liu et al., 2019). We chose the hyperparameters that did best for the baseline QA model (without decompositions) on our dev set: batch size 64, learning rate 1.5×10^{-5}, and weight decay 0.01. Similarly, for BERT experiments, we fix the number of epochs to 2 and choose hyperparameters by sweeping over the recommended ranges from Devlin et al. (2019) for learning rate ($\{2 \times 10^{-5}, 3 \times 10^{-5}, 5 \times 10^{-5}\}$) and batch size ($\{16, 32\}$). For $\text{BERT}_{\text{BASE}}$, we thus choose learning rate 2×10^{-5} and batch size 16, and for $\text{BERT}_{\text{LARGE}}$, we use the whole-word masking model with learning rate 2×10^{-5} and batch size 32. $\text{RoBERTA}_{\text{LARGE}}$ and $\text{BERT}_{\text{LARGE}}$ have 340M parameters, while $\text{BERT}_{\text{BASE}}$ has 110M parameters. We train all QA models with mixed precision floating point arithmetic (Micikevicius et al., 2018), distributing training across 8, 32GB NVIDIA V100 GPUs, lasting roughly 6 hours.

Dataset	Question and ONUS Decomposition
FEVER 1.0	**Q1**: The highest point of the Hindu Kush is Everest. **SQ**$_1$: The highest point of the Hindu Kush? **SQ**$_2$: Where is Everest? **Q2**: John Dolmayan was born on July 15, 1873. **SQ**$_1$: When was John Dolmayan born? **SQ**$_2$: Who was born on July 15, 1873.? **Q3**: Colin Kaepernick became a starter during the 49ers 63rd season in the Republican party. **SQ**$_1$: When did Colin Kaepernick become a starter? **SQ**$_2$: The 49ers 63rd season in the Republican party.? **Q4**: Buffy Summers has been written by Sarah Michelle Gellar. **SQ**$_1$: When has Buffy Summers been written? **SQ**$_2$: Who was Sarah Michelle Gellar.?
FEVER 2.0	**Q1**: Brad Wilk co-founded Rage with Tom Morello and Zack de la Rocha before 1940. **SQ**$_1$: When did Brad Wilk co-founded Rage with Tom Morello? **SQ**$_2$: Who was Zack de la Rocha before 1940? **Q2**: David Spade starred in a 2015 American comedy film directed by Fred Wolf **SQ**$_1$: When was David Spade born? **SQ**$_2$: Who directed the 2015 American comedy film? **Q3**: Java is in Indonesia and was formed by volcanic eruptions Pleistocene Era. **SQ**$_1$: Where is Java in Indonesia? **SQ**$_2$: When were the last volcanic eruptions of Pleistocene Era. **Q4**: Henry Cavill played a fictional character, a superhero appearing in American comic books published by DC Comics. **SQ**$_1$: When did Henry Cavill play a fictional character? **SQ**$_2$: Who are the American superhero appearing in American comic books?
CLEVR	**Q1**: How many cubes are small brown objects or rubber things? **SQ**$_1$: How many cubes are small? **SQ**$_2$: What are brown objects or rubber things? **Q2**: What material is the small ball that is in front of the big metal cylinder behind the block that is to the left of the small yellow rubber sphere made of? **SQ**$_1$: What material is the small ball? **SQ**$_2$: The big metal cylinder behind the big metal cylinder is to the left of the small yellow rubber sphere made of? **Q3**: There is a object in front of the large cyan rubber thing; what is its material? **SQ**$_1$: Why is there a object in front of the large cyan rubber thing? **SQ**$_2$: What is its material? **Q4**: Are there any other things that have the same material as the yellow thing? **SQ**$_1$: Where are there any other things that have the same material? **SQ**$_2$: The yellow thing?
Complex Web Questions	**Q1**: What is the major religions in UK that believes in the deities "Telangana Talli"? **SQ**$_1$: What is the major religions in UK? **SQ**$_2$: Who believes in the deities "Telangana Talli"? **Q2**: Where to visit in Barcelona that was built before 1900? **SQ**$_1$: Where to visit in Barcelona? **SQ**$_2$: What was built before 1900? **Q3**: The person who wrote the lyrics for "Dirge for Two Veterans" was influenced by what? **SQ**$_1$: The person who wrote the lyrics? **SQ**$_2$: What was the influence of "Dirge for Two Veterans"? **Q4**: What country with Zonguldak province as its second division speaks Arabic? **SQ**$_1$: What country with Zonguldak province as its second division? **SQ**$_2$: Who speaks Arabic?

Table 7: **Zero-shot Unsupervised Decompositions** of questions or claims from other datasets using our ONUS model trained on HOTPOTQA and Common Crawl questions (without further, dataset-specific fine-tuning).

Q: What is the name of the singer who's song was released as the lead single from the album "Confessions" and that had popular song stuck behind for 8 consecutive weeks?	
Variable ONUS+ FastText	**Sub-Q1** What is the name of the singer? **Sub-Q2** Who's the song was released as the lead single from the album? **Sub-Q3** What popular song was stuck behind for 8 consecutive weeks?
ONUS+ FastText	**Sub-Q1** What is the name of the singer? **Sub-Q2** What song was released as the lead single from the album "Confessions," and that had popular song stuck behind for 8 consecutive weeks?
ONUS+ Random	**Sub-Q1** What is the name of the singer who's song was released as the lead single from the album "Confessions? **Sub-Q2** Why did popular song stuck behind for 8 consecutive weeks?
Seq2Seq+ FastText	**Sub-Q1** What is the name of the lead singer of the group? **Sub-Q2** When was the last time you heard a song called "on and on" by a female vocalist?
Seq2Seq+ Random	**Sub-Q1** What is the maximum number of students allowed to take part in the program? **Sub-Q2** What is the maximum number of students allowed per year?
PseudoD + FastText	**Sub-Q1** Which single was released as the album's lead single? **Sub-Q2** Who did the song writing, what are some of the stories behind the songs?
PseudoD + Random	**Sub-Q1** What was sold to foreign ptts? **Sub-Q2** What tends to lead to less money?
DecompRC	**Sub-Q1** Which singer? **Sub-Q2** What is the name of alicia keys's song was released as the lead single from the album "Confessions", and that had popular song stuck behind for 8 consecutive weeks?

Table 8: Various decomposition methods for the question *"What is the name of the singer who's song was released as the lead single from the album "Confessions," and that had popular song stuck behind for 8 consecutive weeks?"* Here, the Variable-Length ONUS model decomposed the question into three subquestions rather than two.

Q: Are both Coldplay and Pierre Bouvier from the same country?	
Variable ONUS	**Sub-Q1** Who are similar musical artists to Coldplay? **Sub-Q2** Where is Pierre Bouvier from?
ONUS+ FastText	**Sub-Q1** Where are Coldplay and Coldplay from? **Sub-Q2** What country is Pierre Bouvier from?
ONUS+ Random	**Sub-Q1** Why are both Coldplay and Pierre Bouvier from the same country? **Sub-Q2** What is the purpose of a speech?
Seq2Seq+ FastText	**Sub-Q1** Who are similar musical artists to Pierre Bouvier? **Sub-Q2** In the same year, Pierre Bouvier, Pierre Bouvier, Pierre Bouvier and Pierre Bouvier?
Seq2Seq+ Random	**Sub-Q1** What is the maximum number of students allowed to take part in the program? **Sub-Q2** What is the maximum number of students allowed to take part in the course of the course of the course of the course of the course of the course of the course of the course of the course of the course of the course of the course of the course of the course of the course of?
PseudoD + FastText	**Sub-Q1** What do a handful of couples have in common with Coldplay, Coldplay, and Pierre Bouvier? **Sub-Q2** Where are the french alps?
PseudoD + Random	**Sub-Q1** In what year did Fermat declare Fermat's little theorem? **Sub-Q2** When did the United States withdraw from the Bretton Woods Accord?
DecompRC	**Sub-Q1** Is Coldplay from which country? **Sub-Q2** Is Pierre Bouvier from which country?

Table 9: Various decomposition methods for: *"Are both Coldplay and Pierre Bouvier from the same country?"*

Q: Who is older, Annie Morton or Terry Richardson?	
Variable ONUS	**Sub-Q1** Who is Annie Morton? **Sub-Q2** Who was Terry Morton?
ONUS+ FastText	**Sub-Q1** Who is Annie Morton? **Sub-Q2** When was Terry Richardson born?
ONUS+ Random	**Sub-Q1** Who is older, Annie Morton? **Sub-Q2** Who is Terry Richardson?
Seq2Seq+ FastText	**Sub-Q1** Who is Terry Morton? **Sub-Q2** Who is Terry Morton?
Seq2Seq+ Random	**Sub-Q1** What is the maximum number of students allowed to take part in the program? **Sub-Q2** What is the maximum number of students allowed to take part in the program?
PseudoD + FastText	**Sub-Q1** Who was Terry Richardson? **Sub-Q2** Who changes positions more, Annie Morton, Cotton Hill, or Terry Richardson?
PseudoD + Random	**Sub-Q1** What did Decnet Phase I become? **Sub-Q2** What group can amend the Victorian constitution?
DecompRC	**Sub-Q1** Annie Morton is born when? **Sub-Q2** Terry Richardson is born when?

Table 10: Various decomposition methods for: *"Who is older, Annie Morton or Terry Richardson?"*