Accessible Abstract: For computers to answer complicated questions online, they often need to put together multiple pieces of information (Ronald Reagan was both governor of California and an actor in Bedtime for Bonzo). However, existing approaches use the links in Wikipedia to combine these clues. This research helps computers find connected information without using these explicit links.

Links:

Downloaded from [http://umiacs.umd.edu/~jbg/docs/2021_naacl_multi_ance.pdf](http://umiacs.umd.edu/~jbg/docs/2021_naacl_multi_ance.pdf)

Contact Jordan Boyd-Graber (jbg@boydgraber.org) for questions about this paper.
Abstract

Complex question answering often requires finding a reasoning chain that consists of multiple evidence pieces. Current approaches incorporate the strengths of structured knowledge and unstructured text, assuming text corpora is semi-structured. Building on dense retrieval methods, we propose a new multi-step retrieval approach (BEAMDR) that iteratively forms an evidence chain through beam search in dense representations. When evaluated on multi-hop question answering, BEAMDR is competitive to state-of-the-art systems, without using any semi-structured information. Through query composition in dense space, BEAMDR captures the implicit relationships between evidence in the reasoning chain. The code is available at https://github.com/henryzhao5852/BeamDR.

1 Introduction

Answering complex questions requires combining knowledge pieces through multiple steps into an evidence chain (Ralph Hefferline → Columbia University in Figure 1). When the available knowledge sources are graphs or databases, constructing chains can use the sources’ inherent structure. However, when the information needs to be pulled from unstructured text (which often has better coverage), standard information retrieval (IR) approaches only go “one hop”: from a query to a single passage.

Recent approaches (Dhingra et al., 2020; Zhao et al., 2020a,b; Asai et al., 2020, inter alia) try to achieve the best of both worlds: use the unstructured text of Wikipedia with its structured hyperlinks. While they show promise on benchmarks, it’s difficult to extend them beyond academic testbeds because real-world datasets often lack this structure. For example, medical records lack links between reports.

Dense retrieval (Lee et al., 2019; Guu et al., 2020; Karpukhin et al., 2020, inter alia) provides a promising path to overcome this limitation. It encodes the query and evidence (passage) into dense vectors and matches them in the embedding space. In addition to its efficiency—thanks to maximum inner-product search (MIPS)—Xiong et al. (2021a) show that dense retrieval rivals BERT (Devlin et al., 2019)-based (sparse) retrieve-then-rerank IR pipelines on single step retrieval. Unlike traditional term-based retrieval, fully learnable dense encodings provide flexibility for different tasks.

This paper investigates a natural question: can we build a retrieval system to find an evidence chain on unstructured text corpora? We propose a new multi-step dense retrieval method to model the implicit relationships between evidence pieces. We use beam search (Section 2) in the dense space to find and cache the most relevant candidate chains and iteratively compose the query by appending the retrieval history. We improve the retrieval by encouraging the representation to discriminate hard negative evidence chains from the correct chains, which are refreshed by the model.

We evaluate Beam Dense Retrieval (BEAMDR) on HOTPOTQA (Yang et al., 2018), a multi-
hop question answering benchmark. When re-
trieving evidence chains directly from the corpus
(full retrieval), Beam Dense Retriever (BEAM DR) is competitive to the state-of-the-art cascade reranking systems that use Wikipedia links. Combined with standard rerank-
ing and answer span extraction modules, the gain from full retrieval propagates to findings answers (Section 3). By iteratively composing the query representation, BEAM DR captures the hidden “semantic” relationships in the evidence (Section 4).

2 BEAM DR: Beam Dense Retriever

This section first discusses preliminaries for dense retrieval, then introduces our method, BEAM DR.

2.1 Preliminaries

Unlike classic retrieval techniques, dense re-
trieval methods match distributed text representa-
tions (Bengio et al., 2013) rather than sparse vec-
tors (Salton, 1968). With encoders (e.g., BERT) to embed query $q$ and passage $p$ into dense vec-
tors $E_Q(q)$ and $E_P(p)$, the relevance score $f$ is computed by a similarity function $sim(\cdot)$ (e.g., dot product) over two vector representations:

$$f(q, p) = sim(E_Q(q), E_P(p)). \quad (1)$$

After encoding passage vectors offline, we can effi-
ciently retrieve passage through approximate nearest neighbor search over the maximum inner product with the query, i.e., MIPS (Shrivastava and Li, 2014; Johnson et al., 2017).

2.2 Finding Evidence Chains with BEAM DR

We focus on finding an evidence chain from an unstructured text corpus for a given question, often the hardest part of complex question answering. We formulate it as multi-step retrieval problem. Formally, given a question $q$ and a corpus $C$, the task is to form an ordered evidence chain $p_1...p_n$ from $C$, with each evidence a passage. We focus on the supervised setting, where the labeled evidence set is given during training (but not during testing).

Finding an evidence chain from the corpus is challenging because: 1) passages that do not share enough words are hard to retrieve (e.g., in Figure 1, the evidence Columbia University); 2) if you miss one evidence, you may err on all that come after.

We first introduce scoring a single evidence chain, then finding the top $k$ chains with beam search, and finally training BEAM DR.

Evidence Chain Scoring The score $S_n$ of evidence chain $p_1, \ldots, p_n$ is the product of the (normalized) relevance scores of individual evidence pieces. At each retrieval step $t$, to incorporate the information from both the question and retrieval history, we compose a new query $q_t$ by appending the tokens of retrieved chains $p_1, \ldots, p_{t-1}$ to query $q$ ($q_t = [q; p_1; \ldots; p_{t-1}]$), we use MIPS to find relevant evidence piece $p_t$ from the corpus and update the evidence chain score $S_t$ by multiplying the current step $t$’s relevance score $f(q_t, p_t) * S_{t-1}$.

Beam Search in Dense Space Since enumerating all evidence chains is computationally impossible, we instead maintain an evidence cache. In the structured search literature this is called a beam: the $k$-best scoring candidate chains we have found thus far. We select evidence chains with beam search in dense space. At step $t$, we enumerate each candidate chain $j$ in the beam $p_{j,1}...p_{j,t-1}$, score the top $k$ chains and update the beam. After $n$ steps, the $k$ highest-scored evidence chains with length $n$ are finally retrieved.

Training BEAM DR The goal of training is to learn embedding functions that differentiate positive (relevant) and negative evidence chains. Since the evidence pieces are unordered, we sample positive permuted evidence chains from the gold evidence set. A negative chain has at least one evidence piece that is not in the gold evidence set. For each step $t$, the input is the query $q$, a sampled positive chain $P^+_t = p^+_1, \ldots, p^+_t$ and $m$ sampled negative chains $P^-_{j,t} = p^-_1, \ldots, p^-_t$. We update the negative log likelihood (NLL) loss:

$$L(q, P^+, P^-_1, ..., P^-_m) = \sum_t e^{f([q; P^+_{t-1}, p^+_t])} \left(\sum_{j=1}^m e^{f([q; P^-_{j,t-1}, p^-_t])}\right)^{-1}. \quad (2)$$

Rather than using local in-batch or term matching negative samples, like Guu et al. (2020) we select negatives from the whole corpus, which can be more effective for single-step retrieval (Xiong et al., 2021a). In multi-step retrieval, we select negative evidence chains from the corpus. Beam search on the training data finds the top $k$ highest scored negative chains for each retrieval step. Since the model parameters are dynamically updated, we asynchronously refresh the negative chains with the up-to-date model checkpoint (Guu et al., 2020; Xiong et al., 2021a).
Table 1: Compare B\(\text{EAM}\)DR with other retrieval systems. Top: Retrieval from the whole corpus, bottom: Reranking from top 100 full retrieval outputs. * indicates parallel work.

### Experiments: Retrieval and Answering

Our experiments are on HOTPOTQA fullwiki setting (Yang et al., 2018), the multi-hop question answering benchmark. We mainly evaluate on retrieval that extracts evidence chains (passages) from the corpus; we further add a downstream evaluation on whether it finds the right answer.

#### 3.1 Experimental Setup

**Metrics** Following Asai et al. (2020), we report four metrics on retrieval: answer recall (AR), if answer span is in the retrieved passages; passage recall (PR), if at least one gold passage is in the retrieved passages; Passage Exact Match (P EM), if both gold passages are included in the retrieved passages; and Exact Match (EM), whether both gold passages are included in the top two retrieved passages (top one chain). We report exact match (EM) and \(F_1\) on answer spans.

**Implementation** We use a \textsc{bert}-base encoder for retrieval and report both \textsc{bert} base and large for span extraction. We warm up \textsc{beamdr} with TF-IDF negative chains. The retrieval is evaluated on ten passage chains (each chain has two passages). To compare with existing retrieve-then-rerank cascade systems, we train a standard \textsc{bert} passage reranker (Nogueira and Cho, 2019), and evaluate on ten chains reranked from the top 100 retrieval outputs. We train \textsc{beamdr} on six 2080Ti GPUs, three for training, three for refreshing negative chains. We do not search hyper-parameters and use suggested ones from Xiong et al. (2021a).

#### 3.2 Passage Chain Retrieval Evaluation

**Baselines** We compare \textsc{beamdr} with TF-IDF, Semantic Retrieval (Nie et al., 2019, SR), which uses a cascade \textsc{bert} pipeline, and the Graph recurrent retriever (Asai et al., 2020, GRR), our main baseline, which iteratively retrieves passages following the Wikipedia hyperlink structure, and is state-of-the-art on the leaderboard. We also compare against a contemporaneous model, multi-hop dense retrieval (Xiong et al., 2021b, MDR).

**Results: Robust Evidence Retrieval without Document Links** Table 1 presents retrieval results. On full retrieval, \textsc{beamdr} is competitive to GRR, state-of-the-art reranker using Wikipedia hyperlinks. \textsc{beamdr} also has better retrieval than the contemporaneous MDR. Although both approaches build on dense retrieval, MDR is close to \textsc{beamdr} with TF-IDF negatives. We instead refresh negative chains with intermediate representations, which help the model better discover evidence chains. Our ablation study (Greedy search) indicates the importance of maintaining the beam during inference. With the help of cross-attention between the question and the passage, using \textsc{bert} to rerank \textsc{beamdr} outperforms all baselines.

**Varying the Beam size** Figure 2 plots the Passage EM with different beam sizes. While initially increasing the beam size improves Passage Exact Match, the marginal improvement decreases after a beam size of forty.

#### 3.3 Answer Extraction Evaluation

**Baselines** We compare \textsc{beamdr} with TXH (Zhao et al., 2020b), GRR (Asai et al., 2020) and the contemporaneous MDR (Xiong et al., 2021b). We use released code from GRR (Asai et al., 2020) following its settings on \textsc{bert} base and large. We use four 2080Ti GPUs.

**Results** Using the same implementation but on our reranked chains, \textsc{beamdr} outperforms GRR.
Table 2: HotPOTQA dev and test set answer exact match (EM) and F1 results. * indicates parallel work.

<table>
<thead>
<tr>
<th>Retriever Reader</th>
<th>Dev EM</th>
<th>F1</th>
<th>Test EM</th>
<th>F1</th>
</tr>
</thead>
<tbody>
<tr>
<td>BERT base Reader</td>
<td>TXH</td>
<td>54.0</td>
<td>66.2</td>
<td>51.6</td>
</tr>
<tr>
<td></td>
<td>GRR</td>
<td>52.7</td>
<td>65.8</td>
<td>-</td>
</tr>
<tr>
<td>BEAMDR</td>
<td>GRR</td>
<td>54.9</td>
<td>68.0</td>
<td>-</td>
</tr>
<tr>
<td>BERT large wwm Reader</td>
<td>GRR</td>
<td>60.5</td>
<td>73.3</td>
<td>60.0</td>
</tr>
<tr>
<td></td>
<td>BEAMDR</td>
<td>61.3</td>
<td>74.1</td>
<td>60.4</td>
</tr>
<tr>
<td></td>
<td>MDR*</td>
<td>61.5</td>
<td>74.7</td>
<td>-</td>
</tr>
<tr>
<td>ELECTRA large Reader</td>
<td>MDR*</td>
<td>63.4</td>
<td>76.2</td>
<td>62.3</td>
</tr>
</tbody>
</table>

Table 3: Passage Recall and overlap comparison between BEAMDR and GRR with different hop passages. Systems with † filter second hop passages with links.

<table>
<thead>
<tr>
<th>Models</th>
<th>Passage Recall</th>
<th>Overlap</th>
</tr>
</thead>
<tbody>
<tr>
<td>GRR</td>
<td>First hop 85.1</td>
<td>85.3</td>
</tr>
<tr>
<td>BEAMDR</td>
<td>Second hop 86.4</td>
<td>78.9</td>
</tr>
<tr>
<td>BEAMDR†</td>
<td>First hop 88.0</td>
<td>87.1</td>
</tr>
</tbody>
</table>

Table 4: We manually analyze 100 bridge questions and categorize model errors.

4.2 Hop Analysis

To study model behaviors under different hops, we use heuristics to infer the order of evidence passages. In Table 3, BEAMDR slightly wins on first hop passages, with the help of hyperlinks, GRR outperforms BEAMDR on second hop retrieval. Only 21.9% of the top-10 BEAMDR chains are connected by links. BEAMDR wins after using links to filter candidates.

4.3 Human Evaluation on Model Errors and Case Study

To understand the strengths and weaknesses of BEAMDR compared with GRR, we manually analyze 100 bridge questions from the HotPOTQA development set. BEAMDR predicts fifty of them correctly and GRR predicts the other fifty correctly (Tables 4 and 5).

Strengths of BEAMDR. Compared to GRR, the largest gain of BEAMDR is to identify question entity passages. As there is often little context overlap besides the entity surface form, a term-based approach (TF-IDF used by GRR) falters. Some of the GRR errors also come from using reverse links to find second hop passages (i.e., the second hop passage links to the first hop passage).

We label the passage that contains the answer as the second hop passage, while the other one as the first hop passage. If both passages include the answer, passage title mentioned in the question is the first hop passage.
Q: Chris Williams last played for which football club from the National League North?

Passage 1: Christopher Jonathan “Chris” Williams is an English semi-professional footballer who last played for Salford City as a forward.

Passage 2: Salford City Football Club is a professional football club in the Kersal area of Salford, Greater Manchester, England.

Table 5: Case study of BEAMDR and GRR retrieval. Term-based retrieval approaches (TF-IDF used by GRR) is unable to distinguish two players with same name. BEAMDR correctly identifies the question entity.

Weaknesses of BEAMDR. Like Karpukhin et al. (2020), many of BEAMDR’s errors could be avoided by simple term matching. For example, matching “What screenwriter with credits for Evolution co-wrote a film starring Nicolas Cage and Téa Leoni?” to the context “The Family Man is a 2000 American film written by David Diamond and David Weissman, and starring Nicolas Cage and Téa Leoni.”.

5 Related Work

Extracting multiple pieces of evidence automatically has applications from solving crossword puzzles (Littman et al., 2002), graph database construction (De Melo and Weikum, 2009), and understanding relationships (Chang et al., 2009; Iyyer et al., 2016) to question answering (Ferrucci et al., 2010), which is the focus of this work.

Given a complex question, researchers have investigated multi-step retrieval techniques to find an evidence chain. Knowledge graph question answering approaches (Talmor and Berant, 2018; Zhang et al., 2018, inter alia) directly search the evidence chain from the knowledge graph, but falter when KG coverage is sparse. With the release of large-scale datasets (Yang et al., 2018), recent systems (Nie et al., 2019; Zhao et al., 2020b; Asai et al., 2020; Dhingra et al., 2020, inter alia) use Wikipedia abstracts (the first paragraph of a Wikipedia page) as the corpus to retrieve the evidence chain. Dhingra et al. (2020) treat Wikipedia as a knowledge graph, where each entity is identified by its textual span mentions, while other approaches (Nie et al., 2019; Zhao et al., 2020b) directly retrieve passages. They first adopt a single-step retrieval to select the first hop passages (or entity mentions), then find the next hop candidates directly from Wikipedia links and rerank them. Like BEAMDR, Asai et al. (2020) use beam search to find the chains but still rely on a graph neural network over Wikipedia links. BEAMDR retrieves evidence chains through dense representations without relying on the corpus semi-structure. Qi et al. (2019, 2020) iteratively generate the query from the question and retrieved history, and use traditional sparse IR systems to select the passage, which complements BEAMDR’s approach.

6 Conclusion

We introduce a simple yet effective multi-step dense retrieval method, BEAMDR. By conducting beam search and globally refreshing negative chains during training, BEAMDR finds reasoning chains in dense space. BEAMDR is competitive to more complex SOTA systems albeit not using semi-structured information.

While BEAMDR can uncover relationship embedded within a single question, future work should investigate how to use these connections to resolve ambiguity in the question (Elgohary et al., 2019; Min et al., 2020), resolve entity mentions (Guha et al., 2015), connect concepts across modalities (Lei et al., 2018), or to connect related questions to each other (Elgohary et al., 2018).

Acknowledgments

We thank the anonymous reviewers and meta-reviewer for their suggestions and comments. Zhao is supported by the Office of the Director of National Intelligence (ODNI), Intelligence Advanced Research Projects Activity (IARPA), via the BETTER Program contract 2019-19051600005. Boyd-Graber is supported by NSF Grant IIS-1822494. Any opinions, findings, conclusions, or recommendations expressed here are those of the authors and do not necessarily reflect the view of the sponsors.
References


