A Semantic Search Pipeline for Causality-driven Adhoc Information Retrieval

Dhairya Dalal[1], Sharmi Dev Gupta[2] and Bentolhoda Binaei[1]

[1]SFI Centre for Research and Training in Artificial Intelligence, Data Science Institute, National University of Ireland Galway

[2]SFI Centre for Research and Training in Artificial Intelligence, School of Computer Science and Information Technology, University College Cork

Abstract

We present a unsupervised semantic search pipeline for the Causality-driven Adhoc Information Retrieval (CAIR-2021) shared task. The CAIR shared task expands traditional information retrieval to support the retrieval of documents containing the likely causes of a query event. A successful system must be able to distinguish between topical documents and documents containing causal descriptions of events that are causally related to the query event. Our approach involves aggregating results from multiple query strategies over a semantic and lexical index. The proposed approach leads the CAIR-2021 leaderboard and outperformed both traditional IR and pure semantic embedding-based approaches.

Keywords

semantic search, causal information retrieval, causality detection, causal search

1. Introduction

The Causality-driven Adhoc Information Retrieval (CAIR) shared task consists of retrieving documents with the likely causes of a query event [1]. The search system must be able to differentiate between topical documents and casual documents. Traditional information retrieval (IR) systems usually rely on keyword matching and corpus level n-gram statistics to score which documents are most topically relevant to a provided query. In contrast, given a query event (e.g. Shashi Tharoor resigned), the goal of the causal search system is to identify documents that contain causal information about the events that lead to the query event. For example, causally relevant documents for the query in Figure 1 would refer to the IPL controversy and illicit behavior by Shashi Tharoor. General documents that mention Shashi Tharoor, while topically relevant, may not be causally relevant if they do not contain information about his misbehavior.

In this paper, we describe our solution for the CAIR shared task. We design a unsupervised semantic search pipeline, which aggregates results across several query strategies and indices. The pipeline leverages both a lexical index and a semantic index to retrieve causally relevant documents. Our approach both outperformed standard IR baselines and semantic baselines and was the top method on the CAIR-2021 task leaderboard.

Forum for Information Retrieval Evaluation, December 13-17, 2021, India

d.dalal1@nuigalway.ie (D. Dalal); sharmi.devgupta@cs.ucc.ie (S. D. Gupta); b.binaei1@nuigalway.ie (B. Binaei)

0000-0003-0279-234X (D. Dalal); 0000-0003-0376-2206 (S. D. Gupta)

Query (title)	Shashi Tharoor resigned
Narrative	Shashi Tharoor got involved in the IPL controversy, in a way that his friend Sunanda Pushkar was given free equity by IPL Kochi and he advised against prying into the consortiums ownership, Lalit Modi disclosed on Twitter. Arguments presented against him, and how these events lead up to his resignation as a Minister of Parliament would be considered as relevant documents. Any gossips related to his personal relationship with Sunanda Pushkar are not considerable.

Figure 1: Example CAIR topic. Each topic consists of a query (title), which describes an event, and a narrative, which contains descriptions of documents that are causally relevant to the event.

2. Related Works

Datta et al. [2] provide a brief survey of the literature on causality in natural language processing and explore the task of causal information retrieval in the context of news articles. They also introduce a recursive causal retrieval model which allows for identifying the causal chain of events that led to a news event. Datta et al. [3] propose an unsupervised pseudo-relevance feedback approach that estimates the distribution of infrequent terms that are potentially relevant to the causality of the query event. Recent advances in IR have focused on neural re-ranking and leveraging latent embeddings to improve the overall recall and semantic relevance of returned results. For example, Pang et al. [4] propose SetRank, a permutation-invariant ranking model that jointly learns the embeddings of retrieved documents using self-attention. Most modern IR approaches combine lexical and semantic approaches. For example, Gao et al. [5] presents CLEAR in which a residual-based learning framework teaches the neural embedding to be complementary to the lexical retrieval model. Our approach follows the trend of combining lexical models with semantic embeddings.

3. Methods

Our approach focused on developing an unsupervised semantic search pipeline. Documents were indexed in two indices: a semantic index and a lexical index (see Section 3.1). Results from multiple queries across the two indices were then aggregated to return the most relevant documents. We additionally explored a post query filter step that aimed to identify documents that contained causal language in the context of the query event. This approach did not produce viable results and was not pursued. In this section, we will present our methodology and experimental setup in further detail.

3.1. Document Indexing

Two document indices were created for our semantic search pipeline. The first was a **lexical index** that treated documents as bags of words and was optimized for Okapi BM25 [6] retrieval.

Before indexing, documents were cleaned and tokenized using standard preprocessing steps: lowercasing, stripping out all non-alphanumeric characters, and lemmatization. Thus each document D was broken into lemmatized unigram tokens $t_1 ... t_n$. Next, the tokenized documents were further processed to support the Okapi BM25 ranking algorithm. Given a query Q which consists of query tokens $q_1, ...q_n$, we score each document D in our index using the following scoring function:

$$Score(D,Q) = \sum_{i=1}^{n} IDF(q_i) \cdot \frac{freq(q_i,D)\cdot(k_1+1)}{freq(q_i,D)+k_1\cdot(1-b+b\cdot\frac{D_{length}}{avgDL})}$$

$IDF(q_i)$ is the inverse document frequency of the query token and $freq(q_i, D)$ is the frequency of query token in the document. Finally, D_{length} is the length of the document (i.e. the total number of tokens) and $avgDL$ is the average document length. Okapi BM25 scores are unbounded and larger scores indicate the retrieved document is more relevant compared to lower scored documents in the context of the query.

The second index was a **semantic index** where documents were represented fixed dimension vector embeddings generated by a sentence embedding model. The broad goal of the semantic index was to retrieve documents that are semantically similar to the query. Semantic relevance is measured by the cosine similarity between query embedding U and document embedding V which can be defined as:

$$similarity(U,V) = \frac{U\cdot V}{\|U\|\cdot\|V\|} = \frac{\sum_{i=0}^{n} U_i V_i}{\sqrt{\sum_{i=1}^{n} U_i^2}\sqrt{\sum_{i=1}^{n} V_i^2}}$$

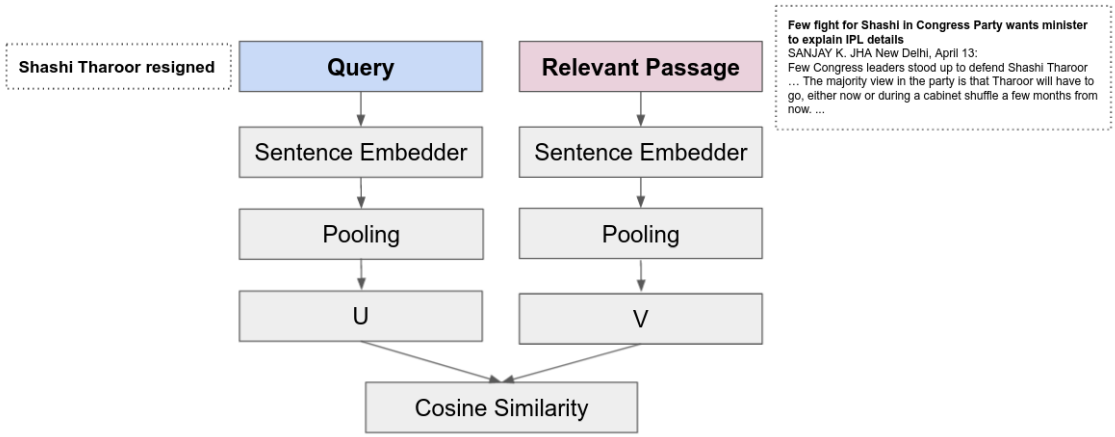

Figure 2: Siamese sentence embedding architecture for asymmetric matching.

Sentence embedding models generate a fixed dimension representation for a provided input text and are trained to represent sentence-level inputs for tasks such as semantic text similarity, paraphrase detection, and textual entailment. Search applications present challenges when considering sentence embedding models. Applying sentence embedding to document-level inputs (e.g. paragraphs or new articles) dilutes the quality of embedding representation and will likely result in poorer performance in the context of dense passage retrieval and ranking. Additionally, there is an input asymmetry challenge where the query length is often shorter

than the relevant document that is to be retrieved. Finally, there may be limited lexical overlap between the query text and the relevant document. As a result standard sentence embeddings models like USE (Universal Sentence Encoder) [7] will struggle for general semantic search use cases. To account for this we use a Siamese network architecture [8] that was pretrained to support asymmetric (Figure 2) matching. The Siamese architecture takes as input query and relevant passage pairs and fine-tunes a shared sentence embedding model to increase the cosine similarity between relevant pairs and decreases the similarity between negative pair samples. The resulting sentence embedding model is better tuned to support the asymmetric nature of determining the semantic similarity between a query and document embedding. Details on the pretrained sentence embedding model can be found in Section 4.2.

3.2. Semantic Search Pipeline

Our semantic search pipeline (Figure 3) aggregates results from three distinct query strategies to produce the final set of relevant causal documents. Provided a topic consisting of a title and narrative (e,g, Figure 1), we treat the title as the query text and narrative as a source for causal keywords.

$Q1$ retrieves the 500 most semantically similar documents from the semantic index. This is accomplished by embedding the query text using the sentence embedding model, retrieving the closest document embeddings based on cosine distance, and then ranking the documents using cosine similarity scores between query embedding and document embedding.

$Q2$ retrieves the 500 most relevant documents from the lexical index, where the relevance is measured by the Okapi BM25 between the candidate documents tokens and query tokens.

$Q3$ also retrieves 500 results from the lexical index but uses causal keywords extracted from the narrative description. The narrative text is first passed through a filter step which removes any statements in the description that describes irrelevant documents. The filter uses a simple keyword-based regex (e.g. not relevant, not considered, irrelevant, etc) to identify those statements. Next, the filtered narrative is converted into a set of keywords using TopicRank [9]. Finally, the causal keywords from the narrative are used to query the lexical index.

$Q1$, $Q2$, and $Q3$ each produce a set of candidate documents ($Q1'$, $Q2'$, and $Q3'$ respectively). These results are sent to an aggregator module that deduplicates and re-ranks all the candidate documents. If a document appears in multiple results sets, its scores are summed. The top 500 documents are returned as the final result set.

3.3. Post Query Causal Filtering

We additionally explored a post query filtering step. This involved extracting causal relations (if any were found) from the candidate document. Candidate documents would have passed this filtering stage if the extracted cause had an overlap with the query text and the extracted effect overlapped with the narrative causal keywords. This approach did not yield promising results on the train topics and was not explored further on the test topics. Often the causal documents did not mention the caused event as the document was reporting news that occurred before the query event. This filtering method would have failed to identify news reports of events in the

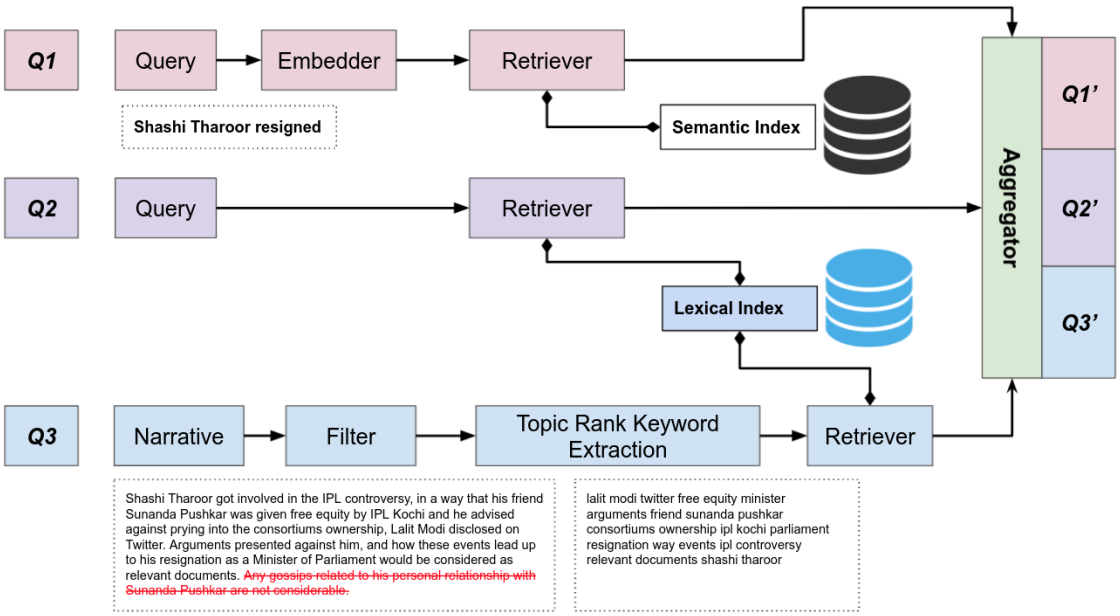

Figure 3: The semantic search pipeline aggregates results from three query strategies, $Q1$, $Q2$, and $Q3$. $Q1$ embeds the query using the sentence embedding model and retrieves the most relevant results based on cosine similarity. $Q2$ and $Q3$ retrieve the most relevant documents from the lexical index. $Q3$ adds filtering and keyword extraction steps to transform the narrative description in causal search terms. Finally results from all three queries ($Q1'$, $Q2'$, and $Q3'$) are aggregated and re-ranked by the aggregator module. The top 500 relevant submissions are returned.

past that lead to the query event because at the time of the reporting, the article did not know about the query event (as it would happen in the future).

4. Experiments

In this section, we describe our implementation and experiment results.

4.1. Data

The CAIR dataset contains 303,291 Telegraph India news articles from 2001 to 2010 [3]. There are 5 train topics and 20 test topics provided. Each topic (e.g. Figure 1) consists of a title, which describes the query event, and a narrative that describes the expected relevant and irrelevant documents.

4.2. Setup

The spacy library [1] was used for prepossessing (i.e. lemmatizing and tokenizing). We used the python rank25 library [2] to implement a lexical index optimized for Okapi BM25 scoring. The default values were used for the $k1$ (1.5) and b (0.75) parameters.

For the semantic index, we use the pretrained *msmarco-distilbert-base-v4* sentence embedding model from the SentenceTransformers library [8]. This model was pretrained on the MS Marco passage ranking dataset [10] which has asymmetric input properties as the query is often shorted the relevant passage. The MS Marco dataset consists of a million queries from the Bing search engine and 8.8 million passages from search results. The passage ranking task requires the model to find the most relevant passages for a provided query and rank them. Documents and qrels from the CAIR corpus were not used for the pretraining of the sentence embedding model.

All the documents in the CAIR corpus were embedded using the *msmarco-distilbert-base-v4* sentence embedding model and then stored in an index optimized for approximate nearest neighbors search. We used the ANNOY python library [3] to store the document embeddings and built a search index of 1000 trees.

4.3. Baselines

We evaluated our approach against four different lexical and semantic baselines. All the baselines returned the top 500 relevant results which were evaluated against the gold document relevance set. Mean Average Precision (MAP) and Precision at 5 (P@5) metrics were used for evaluation. The first (Narrative Only Okapi BM25) baseline used returned results from the lexical index using the narrative text as the query. The second baseline (Query Only Okapi BM25) used the title as the query for lexical index. The third baseline (Query + Narrative Semantic) combined the query and title texts and retrieved the most relevant semantic results from the semantic index. Finally, the last baseline only used the title text to query the semantic index.

4.4. Results

Experiment results can be found in Table 4.4. In addition to our baselines, we include the results of the best submission from the NITS team in the CAIR 2021 shared task. The test set contained 20 topics and a gold relevance set which identified causally relevant documents in the corpus. Our semantic search pipeline outperforms all the baseline methods and leads the shared task leader board. The semantic search pipeline posted a twenty-five percent increase in MAP and a fourteen percent increase in P@5 over the Narrative Only Okapi BM25 baseline.

Our semantic search pipeline uses the same lexical and semantic indexes as the baselines. However, the pipeline is better able to combine the lexical and semantic results to produce the most causally relevant documents. The aggregator module conceptually functions as an ensemble model and weights documents that appear in multiple query result sets higher. Each

[1] https://spacy.io
[2] https://github.com/dorianbrown/rank_bm25
[3] https://github.com/spotify/annoy

Table 1
Experiment Results for the 20 test topics.

Method	MAP	P@5
Semantic Pipeline	.5761	.7800
Narrative Only Okapi BM25 baseline	.3285	.6399
Query Only Okapi BM25 baseline	.2561	.4999
Query + Narrative Semantic baseline	.2239	.5500
Query Only Semantic baseline	.1611	.5000
NITS-Run	.1063	.4800

query strategy utilized information from the topic differently and the final result set reflected that.

Amongst the baselines, the Narrative Only Okapi BM25 baseline was the strongest. The narrative text contains the most useful information about what caused the query event and was expected to provide the best results amongst the baselines. However, the narrative input with a lexical index is still prone to returning topical documents that are not causally relevant. Figure 4 provides a qualitative comparison between the Narrative Only Okapi BM25 baseline and the Semantic Search Pipeline. The baseline models match on terms present in the narrative but the article is focused on accusing Modi of misconduct in the context of the IPL Kochi scandal. In contrast, the Semantic Search Pipeline correctly identifies a document that describes why Shashi Tharoor resigned in relation to scandal and his friend Sunanda Pushkar.

Figure 4: Example results returned by Semantic Search Pipeline and the Narrative Only Okapi BM25 baseline. The baseline returns a topically relevant result based on keyword matches but fails to describe why Shashi Tharoor resigned.

5. Conclusion

In this paper, we introduced a semantic search pipeline for the CAIR-2021 shared task. Our approach aggregated results from multiple query strategies across a lexical and semantic index.

The semantic search pipeline outperformed the lexical and simple semantic baselines and was the top method on the CAIR 2021 leader board. This approach should serve as a stepping stone toward better causal information retrieval. Future work could explore developing a better model of causality and retrieving results using the query title only. The narrative text provides strong clues as the causal terms that would be in the causally relevant documents. A causal search system would have a better way identify and causally linking events.

Acknowledgments

This work was supported by Science Foundation Ireland under grants SFI/18/CRT/6223 (Centre for Research Training in Artificial Intelligence).

References

[1] S. Datta, D. Ganguly, D. Roy, D. Greene, C. Jochim, F. Bonin, Overview of the causality-driven adhoc information retrieval (cair) task at fire-2020, in: Forum for Information Retrieval Evaluation, FIRE 2020, Association for Computing Machinery, New York, NY, USA, 2020, p. 14–17. URL: https://doi.org/10.1145/3441501.3441513. doi:10.1145/3441501.3441513.

[2] S. Datta, D. Greene, D. Ganguly, D. Roy, M. Mitra, Where's the why? in search of chains of causes for query events, in: AICS, 2020.

[3] S. Datta, D. Ganguly, D. Roy, F. Bonin, C. Jochim, M. Mitra, Retrieving potential causes from a query event, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 1689–1692.

[4] L. Pang, J. Xu, Q. Ai, Y. Lan, X. Cheng, J. Wen, Setrank: Learning a permutation-invariant ranking model for information retrieval, in: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval, 2020, pp. 499–508.

[5] L. Gao, Z. Dai, T. Chen, Z. Fan, B. V. Durme, J. Callan, Complement lexical retrieval model with semantic residual embeddings, in: European Conference on Information Retrieval, Springer, 2021, pp. 146–160.

[6] S. Robertson, S. Walker, S. Jones, M. Hancock-Beaulieu, M. Gatford, Okapi at trec-3, in: TREC, 1994.

[7] D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. St. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, B. Strope, R. Kurzweil, Universal sentence encoder for English, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Association for Computational Linguistics, Brussels, Belgium, 2018, pp. 169–174. URL: https://aclanthology.org/D18-2029. doi:10.18653/v1/D18-2029.

[8] N. Reimers, I. Gurevych, Sentence-bert: Sentence embeddings using siamese bert-networks, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, 2019. URL: http://arxiv.org/abs/1908.10084.

[9] A. Bougouin, F. Boudin, B. Daille, TopicRank: Graph-based topic ranking for keyphrase extraction, in: Proceedings of the Sixth International Joint Conference on Natural Language

Processing, Asian Federation of Natural Language Processing, Nagoya, Japan, 2013, pp. 543–551. URL: https://aclanthology.org/I13-1062.

[10] T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Majumder, L. Deng, MS MARCO: A human generated machine reading comprehension dataset, CoRR abs/1611.09268 (2016). URL: http://arxiv.org/abs/1611.09268. arXiv:1611.09268.