Evidence Retrieval for Causal Questions Using Query Expansion And Reranking
Notebook for the Touché Lab on Argument and Causal Retrieval at CLEF 2023

Aron Gaden, Niklas Rausch, Bruno Reinhold and Lukas Zeit-Altpeter

Abstract
We present our runs to the Touché Lab on Argument and Causal Retrieval at CLEF 2023, aiming to retrieve relevant documents for given causal questions. Our approaches rely heavily on query expansions. Both a transformer-based and a technique using expansions using prior knowledge on the stated relationships are used. The retrieved documents are then retrieved using the ChatNoir search engine before being subjected to a simple reranking scheme, ranking documents higher the earlier the causal relationship in question is mentioned.

Keywords
information retrieval, query expansion, reranking, touche

1. Introduction
In this paper, we explain our approach to evidence retrieval for causal questions, as an entry for shared task 2 of Touché at CLEF 2023[1]. Submissions were made via TIRA[2]. Evidence retrieval concerns itself with identifying whether a causal relationship between two events or actions exist. Currently, search machines do not recognize such queries, nor do they retrieve those kinds of relationships. We approached the topic under the aspects of query analysis, query expansion, and document scoring. The goal of the shared task is to retrieve and rank documents by relevance to the topic and detect the document “causal” stance, that is “for”, “against”, or neutral. In our case, we opt to not determine a stance, and only rank documents based on whether they make a claim about possible causal relationships. We (Team He-Man) propose three methods for obtaining causal relationships. The first run (no-expansion) is to be considered a baseline, using only the task’s query as search input. The second run (gpt-expansion) uses responses from manual prompts to ChatGPT as expansions for each query. The third run (causenet-expansion) tries to find terms similar to the search terms in the CauseNet corpus [3]. In this paper, we explain the general methods we applied, as well as attempts we made during development. Whenever it is not obvious, we mention which run the information applies to. Finally, we discuss possible shortcomings and improvements of our approaches.

Our code is available at https://git.uni-jena.de/na93xek/he-man-causal-retrieval
2. Query Preprocessing

Finding documents that confirm or refute causal relations between two entities, forming part of the query, is a stricter constraint than just finding any documents containing those entities. Hence, we attempted to connect both cause and effect with some entity in every query. Broadly speaking, this process consists of two steps: the extraction of the two entities for which the causal relation is to be found from the tokenized queries, and what the relation between them is (i.e. which is cause and which effect). Before analyzing the terms of the query, we split each tokenized query in two parts. This was facilitated by the design of the tasks, which were all following the formula of “Does A cause B?” or similar, where the cause and effect would be divided by some kind of connection structure.

Sentences like “Is there a causal relation between A and B?” were not part of the dataset, and therefore not considered in our processing algorithm. For tokenizing the queries, we used the NLTK python package\(^2\). We extracted all the dividing patterns found in the dataset for this initial split of the query. Here, we disambiguated active and passive patterns. Active patterns (e.g. “lead to”) form part of a query with the structure “A causes B?”, while passive patterns (e.g. “result of”) reverse the relation: “B causes A?”. The data set with 50 questions contained 11 patterns - 7 active and 4 passive patterns. With these patterns, we could exhaustively and correctly split the queries of the data set. After splitting the query in two parts and extracting the relation between the entities, we had to identify the entities in the query parts. In an initial experiment, we queried Wikidata\(^3\) for each n-gram contained within the partial query for every n from 1 to the length of the partial query. We assumed the largest substring of the query that returned a result from the Wikidata query as cause or effect, respectively.

We were able to identify cause and effect correctly in 33 of the 50 tasks, where the cause and effect were individually extracted correctly 41 and 44 times, respectively. Often times, this approach would leave out important context of the query term, such as increased insurance premiums being identified as insurance or high blood pressure medication as high blood pressure. Semantically, these differ drastically from the actual search term.

Hence, we used another solution with a different approach. Using Stanza \[^4\], we built and analyzed the constituency tree of the query. With the knowledge of having two relevant search terms, we could still split the query into two parts. We assigned the longest noun phrase in each query part to be the relevant query term. Using the aforementioned rule-based strategy for identifying cause and effect, we were able to automatically recognize all cause and effect pairs in the data set correctly, hence all he-man runs used the latter method.

3. Query Expansion

3.1. ChatGPT Expansion

GPT stands for “Generative Pre-trained Transformer” \[^5\]. It refers to a type of artificial neural network architecture that is widely used in natural language processing tasks, such as text generation, translation, understanding or expansion of existing text. The “generative” aspect

\(^2\)https://www.nltk.org/

\(^3\)https://www.wikidata.org
of GPT refers to the model’s ability to generate new text based on the input it receives. It can produce coherent and contextually relevant text that resembles human language. However, one needs to keep in mind that ChatGPT occasionally generates inaccurate or nonsensical information. Furthermore, "Pre-trained" means that the model is trained on a large dataset before being fine-tuned for specific tasks. In the case of GPT, it is trained on a massive amount of text data, mainly from a dataset of websites called Common Crawl. This pre-training allows the model to learn patterns, grammar, vocabulary, and other linguistic features from the data. Additionally, ChatGPT has been moderated in order to make the interaction with the AI-tool more natural and safer for humans to use. "Transformer" refers to the specific architecture used in GPT. Transformers are deep learning models that rely on self-attention mechanisms to process input data. They are known for their effectiveness in capturing long-range dependencies in sequences, making them suitable for tasks involving natural language.

The chat tool has been instructed to expand each of the 50 causal queries. All the original queries are interrogative sentences. The cause-and-effect relationship is clearly identifiable by a causal phase, such as e.g. “leads to” or “caused by”. Furthermore, the queries can be posed in an active- or passive voice. The exact prompt is stated in figure 2.

The accuracy of the generated expansions is satisfactory in the way that the generative AI-tool follows the provided structure and generates correct and non-repetitive expansions. However, the amount of generated extension varies between four and seven. Moreover, the process of obtaining the query expansions was quite laborious. This was due to the limited and varying output size and the unsteady availability of the online tool at the time. Each query was expanded individually and composed together afterwards by concatenating the query field of each expansion returned as JSON.
Expand the queries by following this json hierarchy. A query can only have one "cause" one "causal phrase" and one "effect" which are extracted from the query in an information retrieval task, in order to find documents containing the causal relationship. Find both expansions that are formulated in a passive and active voice. Try to use causal phrases like: "causes, results in, leads to, effect of, etc. Try to find at least 5 query expansions. the newly generated queries should be questions too. Only respond in the specified json format!

Figure 2: Prompt to ChatGPT for generation of query expansions

3.2. CauseNet Expansion

A second way of expanding the queries derived from the given topics consists of using prior knowledge on causal questions, to improve the original query. For this we used the CauseNet dataset [3] that contains sources on causal relations between given concepts. We used the CauseNet-Precision variant, featuring around 200,000 relations between 80,000 different concepts along with snippets and IDs of pages in the ClueWeb12 corpus, stating the supposed causal links.

We faced the challenge that not all cause-effect-pairs had exact matches within the dataset. To find matching pairs, we embedded all CauseNet-concepts alongside our extracted terms from the topics using the BERT-Uncased model [7] and performed the matching by choosing those CauseNet-pairs that had the lowest mean cosine-distance to the given cause-effect-pair generated from the topics. The fulltext for the sources given for the matched relation were then retrieved using the ChatNoir-Cache [8]. We extracted the top 5 terms across all sources. This was done by first tokenizing the relevant documents using the pre-trained Punkt tokenizer provided by the NLTK python package 4. The tokens were then converted to lowercase letters before counting occurrences across all documents.

These terms were then added to the original topic to form the expanded query.

The topic is child labor an effect of poverty? was thus expanded with the terms child labour children education work 5, highlighting the role of education in prevention child labour. The complete query created by query expansion is thus: Is child labor an effect of poverty? child labour children education work.

4. Reranking

Across all three runs one query per topic was submitted to the ChatNoir search engine [8]. The top 50 document ids were retrieved alongside their full text before being subjected to a custom reranking scheme. For each sentence in the retrieved document, we then detected if a postulation of the given causal relation was present. The method used here was analogous

4https://www.nltk.org/api/nltk.tokenize.punkt.html
5The duplicate of child labour in both original topic and expansion terms is due to the word-based tokenization that was used when extracting the sources’ top words.
Table 1
All runs submitted for the retrieval task ranked by normalized discounted cumulative gain.

<table>
<thead>
<tr>
<th>Team</th>
<th>Run</th>
<th>NDCG@5</th>
<th>95% confidence interval</th>
</tr>
</thead>
<tbody>
<tr>
<td>He Man</td>
<td>heman_no_expansion_rerank</td>
<td>0.657</td>
<td>[0.564, 0.740]</td>
</tr>
<tr>
<td>Puss In Boots</td>
<td>puss-in-boots_baseline</td>
<td>0.585</td>
<td>[0.503, 0.673]</td>
</tr>
<tr>
<td>He Man</td>
<td>heman_gpt_expansion_rerank</td>
<td>0.374</td>
<td>[0.284, 0.469]</td>
</tr>
<tr>
<td>He Man</td>
<td>heman_causenet_expansion_rerank</td>
<td>0.268</td>
<td>[0.172, 0.368]</td>
</tr>
</tbody>
</table>

to the method described in section 2. We then counted the number of mentions of the given relations, weighing each occurrence by its position within the document with a weight of 1 indicating the very first sentence of the document and the weight getting closer to 0 the further back the occurrence is located. All weights were then summed to get the document’s occurrence score.

ChatNoir returns scores for each retrieved document. The original score and the occurrence score were combined using the following formula:

$$score_{reranked} = score_{ChatNoir} + \frac{score_{occurrence}}{2}$$

We experimented around with different approaches for combining the scores but came to the conclusion to use the exponent-based method to make sure that our score has a sufficient impact on the ranking. 1 is added to the score to make sure that those documents that do not contain and explicit mention are still ranked highly if ChatNoir returns a high enough score.

5. Evaluation

In total we submitted three different runs. Our baseline run by submitting the topics’ titles to ChatNoir before reranking as described above. The ChatNoir and ChatGPT runs additionally expanded the title as described above before retrieving and reranking the results.

Overall, our baseline run achieved the best NDCG across all submitted runs with a NDCG of 0.657 (cf. Table 1). Both expansion techniques significantly lowered the retrieval performance.

6. Conclusion

We have been successful in using constituency parsing and simple rules to identify cause and effect in all queries of the data set. Nevertheless, the approach was engineered for this particular data set and will not perform to this standard in general application. Future causal retrieval models may want to improve on this procedure.

The expansion of the queries with the help of ChatGPT worked well. Although the generation of the expansions was quite manual labor intensive due to the lack of an API. With regards to the CauseNet based expansion it should be noted that while most of the given cause-effect-combinations were successfully matched to a fitting counterpart, some cases failed, resulted in nonsense expansions. The pair of "drinking wine" and "blood urine" was for example matched
to the pair of "eating food" and "diarrhea", resulting in an expansion with the top terms of food, pork, peanut, milk, health.

The evaluation of retrieval results show that both expansion runs submitted by us do not improve the retrieval performance compared to the run solely using our custom reranking scheme. This might be due to a too high number of ill-fitting expansions for the CauseNet run and too long queries formulated by the ChatGPT expansion.

The scoring algorithm can be considered a basic measure. It works similarly to the query analysis method. Since the search corpus is much more complicated and varied than the task queries, we struggled to find many matches in the found documents. This means our proposal leans more towards a high-precision, low-recall solution. Still, the scoring lead to an improvement regarding the retrieval performance in relation to the other baseline run submitted.

References


