RESEARCH ARTICLE

Informative Causality Extraction from Medical Literature via Dependency-Tree–Based Patterns

M. Ahsanul Kabir[1] · AlJohara Almulhim[1] · Xiao Luo[2] · Mohammad Al Hasan[1]

Abstract

Extracting cause-effect entities from medical literature is an important task in medical information retrieval. A solution for solving this task can be used for compilation of various causality relations, such as causality between disease and symptoms, between medications and side effects, and between genes and diseases. Existing solutions for extracting cause-effect entities work well for sentences where the cause and the effect phrases are name entities, single-word nouns, or noun phrases consisting of two to three words. Unfortunately, in medical literature, cause and effect phrases in a sentence are not simply nouns or noun phrases, rather they are complex phrases consisting of several words, and existing methods fail to correctly extract the cause and effect entities in such sentences. Partial extraction of cause and effect entities conveys poor quality, non-informative, and often, contradictory facts, comparing to the one intended in the given sentence. In this work, we solve this problem by designing an unsupervised method for cause and effect phrase extraction, PATTERNCAUSALITY, which is specifically suitable for the medical literature. Our proposed approach first uses a collection of cause-effect dependency patterns as template to extract head words of cause and effect phrases and then it uses a novel phrase extraction method to obtain complete and meaningful cause and effect phrases from a sentence. Experiments on a cause-effect dataset built from sentences from PubMed articles show that for extracting cause and effect entities, PATTERNCAUSALITY is substantially better than the existing methods—with an order of magnitude improvement in the *F*-score metric over the best of the existing methods. We also build different variants of PATTERNCAUSALITY, which use different phrase extraction methods; all variants are better than the existing methods. PATTERN-CAUSALITY and its variants also show modest performance improvement over the existing methods for extracting cause and effect entities in a domain-neutral benchmark dataset, in which cause and effect entities are nouns or noun phrases consisting of one to two words.

✉ Mohammad Al Hasan
alhasan@iupui.edu

1   Department of Computer Science, Indiana University Purdue University Indianapolis, Indianapolis, IN, USA

2   Department of Computer Information and Graphics Technology, Indiana University Purdue University, Indianapolis, IN, USA

1 Introduction

Medical research is advancing at a rapid pace causing information overload in the medical literature. To overcome this challenge and to be up-to-date about the latest discoveries, medical communities extract health-related facts, hypotheses, and real-life evidences from various textual sources, including scientific reports, clinical notes, and online health forums. For this purpose, they use various text mining and information extraction methodologies. Among these, causality extraction is one of the most important extraction methodologies due to the fact that causal sentences are generally used for describing medical facts and hypotheses. For instances, causal sentences are commonly used in medical text for providing the causes of a disease. For instance, the sentence , "diabetes is caused by the absence of insulin secretion, and vitamin D deficiency," reflects disease causality, clearly denoting two of the causes of the diabetes disease. Effectively extracting disease causality [1–3] from such sentences provides knowledge which may help prevention of that disease. Causal sentences can also help discovering the effectiveness of a medical treatment or the side effects associated with the treatment [4, 5]. For example, the sentence, "ACE Inhibitors are used to treat heart-related conditions, but can also cause chronic cough" [6], reflects side effect causality. From such sentences if the cause and the effect phrases can be extracted automatically, researchers can derive facts, build relevant hypotheses, and enrich the medical knowledge-base effectively and efficiently.

Due to the importance of causality extraction in medical domains, several works [7–10] have been proposed, which either identify causal relations from sentences or extract cause and effect pairs to build causal networks. These existing works either use a set of known syntactic patterns to extract the cause and effect entities or use machine learning models to identify those entity pairs. Considering the cause and effect pairs as name entities, existing methods focus on entity extraction, and they performed well when the causes and effects are name entities, or noun phrases, such as the name of diseases, medications, or genes. However, in medical text, the cause or the effect phrases can often be complex, including not only a name entity but also a multi-word phrase consisting of noun or pronouns preceding with modifiers, or even a dependent clause. In those scenarios, the existing methods fail to extract complete phrases, often returning only head words, which is not enough to capture the intended causality. In fact, such a shortcoming can, sometimes, alter the intended causality information completely. For example, consider the sentence "Deficiency in Vitamin D can cause increased mortality rate in Covid-19 patients." One of the existing methods, which we refer to as Logical-Rule Based method [11], extracts "Vitamin D" as the cause term and "increased mortality rate" as the effect term, seemingly conveying the erroneous fact that "Vitamin D causes increase mortality rate"—whereas, it is not "Vitamin D," but the "deficiency of Vitamin D" is the culprit. Another existing method, which we refer to as Word Vector Mapping Based method [12] extract "Deficiency" as the cause term and "increase mortality rate" as the effect term, leaves the readers dumbfounded with the question, deficiency of what? Given the abundance of longer cause and effect phrases in the medical text, a new causality extraction method

is needed which is better at capturing the complete cause and effect phrases, thus retaining the integrity of the fact intended in a causal sentence.

To overcome this difficulty, in this work, we propose a simple, yet highly effective, causality extraction method, which is particularly suited for extracting causality from scientific documents, where the cause and effect terms are generally longer. Our proposed method uses a dependency tree parser and then utilizes a collection of cause-effect dependency patterns for extracting the cause and effect nodes in the dependency tree of a sentence. Then it uses a novel phrase extraction method to obtain complete cause and effect phrases from that sentence. Experiments on a corpus built from PubMed articles show that PATTERNCAUSALITY can successfully extract long cause and effect phrases whereas existing methods fail to do so. As for the previous example, PATTERNCAUSALITY is able to extract "deficiency in Vitamin D" as the cause term of the given sentence.

2 Related Work

Causal relation extraction tasks can be mainly categorized into unsupervised [13, 14], supervised [15, 16], and hybrid approaches [12, 17, 18]. The unsupervised approaches are mainly pattern-based approach, which use causative verbs, causal links, and causal relations between words or phrases to extract cause effect pairs. The supervised approaches need to have a training set with cause and effect phrase pairs labeled, then supervised learning models can be trained to extract the causal relationships between the phrases. Do et al. [19] developed a minimal supervised approach based on constrained conditional model framework with an objective function that takes discourse connectives into consideration. Dasgupta et al. [15] used word embedding with selected linguistic features to construct the representation of entities as input of a bidirectional Long-Short Term Memory (LSTM) model to predict causal entity pairs. Nguyen et al. [20] utilized pre-trained word embedding to train convolution Nerual Network (CNN) to classify given casual pairs. Peng et al. [21] presented a model-based approach that utilizes deep learning architecture to classify relations between pairs of drugs and mutations, and also triplets of drugs, genes, and mutations with N-ary relations across multiple sentences extracted from PubMed corpus. Zhao et al. [9] developed a semi-supervised approach which leverages both contextual information and graph structure to extract unseen causal pairs among multiple sentences within medical text. The known causal pairs were determined using context vector that is calculated conducting TF-IDF on concatenating the words before, after, and between the causal pairs, and the differences between the causal pair embeddings that are generated using Word2Vec. The hybrid approaches use both patterns as well as supervised models to extract causality. One of the earliest hybrid approaches is proposed by Sorgente [11] which has two phases: lexico-syntactic patterns extraction and machine learning model classification.

Text mining and machine learning approaches have been employed in the medical domain for causal relation extraction. Atkinson et al. [7] worked on biomedical texts causal pattern discovery using Bayesian net, but the main focus of this work is causal relation classification; for instance, classifying between harmful and beneficial causal relations. Lee et al. [8] investigated disease causality extraction using lexicon-based

causality term strength and frequency-based causality strength. In this work, the causality relations are defined by a set of terms, such as "causes" and "effects," with associated strength. An et al. [10] explored extracting causal relations using syntactic analysis with word embedding. They used designed syntactic patterns to first obtain triples including the cause and effect pairs, verb that links the pairs, and a binary term denoting whether the relation is passive or negative. Then, word embedding of the verbs in the designed syntactic patterns was used to discover additional verbs that define the causal relations.

3 Materials and Methods

Given a sentence $\mathcal{S}$, which denotes causality between two phrases, hereby called as cause phrase, u and effect phrase, w, our task is to extract the phrases u and w, as completely as possible. We assume that the sentences are from scientific literature where the cause and effect phrases are generally longer. To extract the cause and effect phrases, we use a collection of cause-effect dependency patterns (will be defined later), denoted as $\mathcal{P}$. These patterns are used as template to extract u and w.

Given a sentence, we first find all its noun phrases by dependency tree parsing. Each of these phrases is a candidate to be the cause phrase or the effect phrase. For every such pair of noun phrases, we then validate whether the pair of phrases fits with one of the dependency patterns. If yes, we extract the phrases as potential cause and effect phrases. However, these phrases may not be complete, rather they could merely be head words of a complete cause or effect phrase; so we extend these phrases by finding sentence segments which are associated with the descendant nodes of the cause and effect phrase nodes in the dependency tree. The main novelty of PATTERNCAUSALITY is twofold: First, the utilization of a collection of dependency patterns for extracting core part of cause and effect phrases; second, the extension of the core part of cause and effect phrases by finding sentence segments associated with the correct dependency tree nodes.

Below we discuss the entire process in four different steps: sentence representation, dependency pattern extraction, pattern matching and core cause-effect phrase extraction, and cause-effect phrase extension.

3.1 Sentence Representation

To identify cause effect phrases from a sentence, the sentence should be represented in a form that can preserve its syntactic structure. In such a representation, the syntactic units in a sentence are isolated, which would make the extraction of the cause and effect phrases easier. Dependency tree parsers, such as Spacy [22], Stanford [23], ClearNLP [24], and LTH [25], can serve this purpose. The output of such a parser is a tree in which each node corresponds to a word or a phrase denoting a syntactical unit. The nodes are also labeled by the parts-of-speech of the word or phrase associated to a node. Dependency among different syntactic units is reflected by tree edges, which are labeled by dependency relations. While most of the dependency parsers generate more or less similar dependency tree, we choose

Spacy [22] for this task due to the following reasons: First, Spacy provides an industrial strength API for broader natural language processing tasks, allowing us to build an end-to-end NLP application, including libraries for tokenization, data import, and visualization, in addition to libraries for building the dependency parser. Second, Spacy is well documented and can easily be used for the dependency tree extraction task. Finally, Spacy is very efficient in terms of execution time. For all of the experiments and examples of this research, we have used Spacy for dependency parsing.

Given a sentence *"Most AE-COPD cases are attributed to bacterial or viral respiratory infections and to both types of microorganisms together"*, Fig. 1 shows the dependency tree of this sentence parsed with Spacy. As can be seen, the vertices of the dependency tree are the words or phrases labeled by parts-of-speech; each edge reflects a dependency relation between the two words associated to the end-points of that edge.

3.2 Dependency Pattern Extraction

Dependency pattern is a linguistic structure which denotes a relationship between entities in a sentence. For denoting cause-effect relationship, there exist several dependency patterns in English literature, and a comprehensive compilation of these patterns is needed for extracting cause-effect entities with high recall. To obtain such a listing of cause-effect dependency patterns, we use a supervised machine learning approach. Before discussing the machine learning method, we provide a formal definition of dependency patterns as below.

3.2.1 Definition of Dependency Pattern

Generally, a pattern template of a semantic relation is associated with words or phrases which exhibit that relation in a sentence. For instance, phrases, such as "caused by" and "attributed to," are phrases which are associated with cause-effect relationship between a pair of entities in a sentence. Dependency patterns are formal representation of such phrases through the use of dependency edges obtained from a dependency tree representation of a cause-effect sentence. For such a sentence, dependency edges are composed of incoming and outgoing causal phrases, causative verbs, causal links, and their parts-of-speech (POS) tag.

For instance, let us consider the sentence in Fig. 1, the dependency edge "attributed→to" is part of a dependency pattern because this edge is associated to "attributed
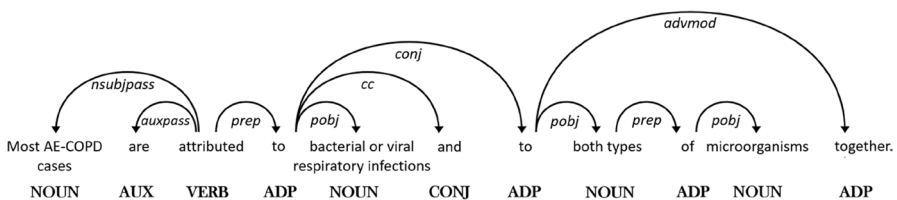

Fig. 1 Dependency tree example for a sentence

to," which denotes a causal relation between the cause phrase "bacterial or viral respiratory infection" and the effect phrase "Most AE-COPD cases." Besides, the POS tags of these words provide specific information about "attributed" and "to," for them being "verb" and "prep" respectively; hence, the POS tags are also included in the dependency edge information. Note that, the word "attribute," in isolation, does not exhibit any cause-effect pattern, whereas dependency edge "attributed→to" is part of a dependency pattern. From this pattern, we can then build a cause-effect pattern template, "Y attributed to X" where X and Y are cause and effect terms, respectively.

3.2.2 Supervised Learning Approach for Dependency Pattern Extraction

Cause-effect pattern extraction can be done manually by experts. However, it is a labor-intensive process; besides, the list of patterns extracted by experts may not cover a large variety of cause-effect sentences. In an earlier work, we developed a supervised learning approach, called ASPER [26], for extracting syntactic patterns associated with any semantic relation. We utilize ASPER for collecting a larger number of cause-effect dependency patterns. Note that, ASPER is used only to find the dependency patterns, but PATTERN-CAUSALITY uses those patterns for extracting cause and effect phrases. In the next paragraph, we briefly illustrate the dependency pattern extraction task performed by ASPER.

Dependency patterns are extracted from a collection of sentences exhibiting cause-effect relation between a cause-effect pair. Different labeled datasets are available in the literature which provide cause and effect terms in sentences, such as ADE [27], Semeval-2007 [28], and Semeval-2010 [29]. Some term-pairs in these datasets do not contain cause-effect relation in their corresponding sentences. These pairs are referred to as negative pairs. For our purpose, ASPER in Fig. 2 is trained to solve a binary classification task to predict whether there is a cause-effect relationship between a given term-pair in a given sentence. The input to ASPER is a collection of edges in the dependency tree representation of the given sentence. The edges are ordered as per shortest path from the cause node to effect node in the dependency tree.

After that, an attention-based bi-directional LSTM model is trained for binary classification. The edges which are important with respect to attention values are then collected to perform frequent itemset mining [30] to collect patterns. Some sentences can exist in a dataset which possesses cause-effect relation but lacks a strong pattern or any pattern at all. In that case, the edges which contain the cause or effect terms get more attention and also frequent itemset mining does not extract any pattern because of infrequency. Below, we provide a formal discussion of the extraction of dependency patterns.

Given a sentence $\mathcal{S} = [w_1, w_2, w_3....w_N]$ with N words or phrases $w_1, w_2....w_N$, let (u, v), where $u, v \in \{w_i\}_{\{1 \leq i \leq N\}}$ is a pair of noun phrases exhibiting cause-effect relation. If $\mathcal{S}$ is parsed with a dependency tree parser, $\mathcal{T} = (V, E)$, each vertex, $v_i \in V$ is associated with a word w_i; besides, the vertices are labeled with the parts-of-speech (POS) tag (e.g., noun, verb, adverb, adjective) of w_i. The edges-set, E, is the set of all directed edges in the dependency tree. Each dependency edge e_{mn} links a parent vertex v_m to a child vertex v_n. We describe the edge e_{mn} as $[w_m, pos(w_m), dep_{mn}, w_n]$, where w_m and w_n are words or phrases associated with v_m and v_n, $pos(v_m)$ denotes the parts of speech tag of w_m and dep_{mn} symbolises the dependency relation between v_m and v_n. For sentence embedding, among all the edges of E, only the shortest path edges between u
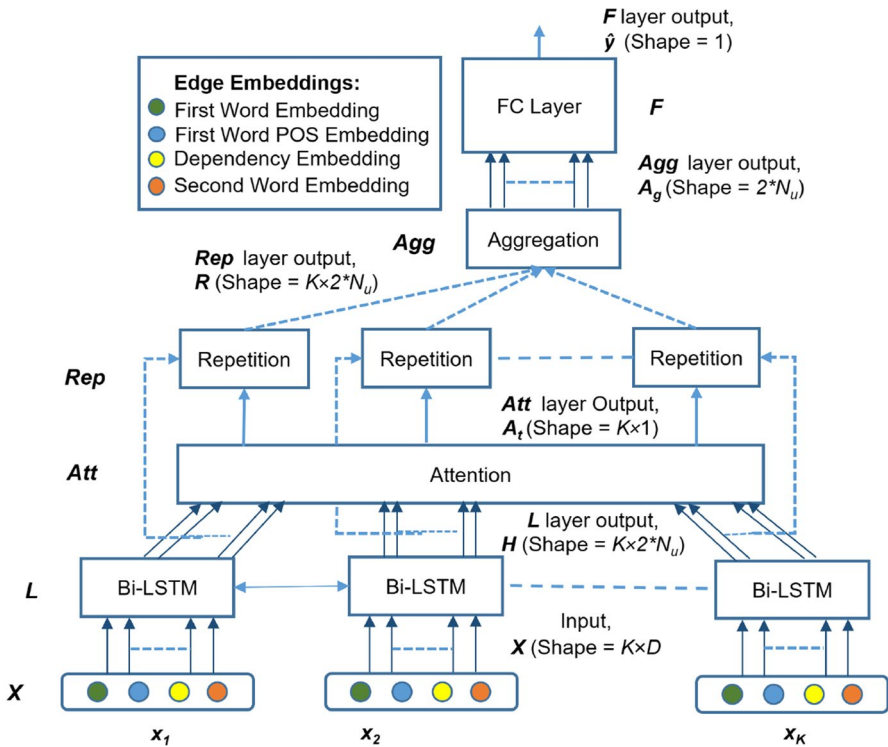
Fig. 2 Description of LSTM with attention layer for binary semantic relationship classification with ASPER model

and v are considered. To explain the embedding of any edge e_{mn}; w_m and w_n are embedded with semantic embedding method, whereas $pos(w_m)$ and dep_{mn} are embedded with one hot embedding. All of these embeddings are then concatenated for edge embedding.

Let for $\mathcal{S}$, there contain K edges, which are embedded as $x_1, x_2, \ldots x_K$. The Bi-LSTM layer of Fig. 2, $\mathcal{L}$, takes x_i as input and outputs two hidden state vectors. The first hidden state vector, $\overrightarrow{h_i}$, is the forward state output, and the second hidden state vector, $\overleftarrow{h_i}$, is the backward state output. Let h_i be the concatenated output of $\overrightarrow{h_i}$ and $\overleftarrow{h_i}$. Also, we define $\mathbf{H}$, which is the concatenation of each h_i output from $\mathcal{L}$ for x_i.

$$\overrightarrow{h_i} = \mathcal{L}(\overrightarrow{h_{i-1}}, x_i), \overleftarrow{h_i} = \mathcal{L}(\overleftarrow{h_{i+1}}, x_i)$$

$$h_i = [\overrightarrow{h_i}, \overleftarrow{h_i}], \mathbf{H} = [h_1, h_2....h_K]$$

Following the Bi-LSTM layer, $\mathcal{L}$, output $\mathbf{H}$ is used as input to the attention layer, **Att**. The attention layer produces, $\mathbf{A}_t$, a vector of size $K \times 1$ where each $a_i \in \mathbf{A}_t$ is a value within a fixed range, $a_i \in [0, 1]$. Each such attention value, a_i, will encode the

relative importance of edge embedding x_i in making the binary classification decision. $\mathbf{A}_t$ is computed as below.

$$Temp = \mathbf{Tanh}(\mathbf{H} * \mathbf{W_1}) * \mathbf{W_2}$$

$$\mathbf{A}_t = \mathbf{Softmax}(Temp)$$

Here, $\mathbf{W_1}$ is a trainable matrix of shape $2 * N_u \times 2 * N_u$, $\mathbf{W_2}$ is another trainable matrix of shape $2 * N_u \times 1$. The shape of temporary variable $Temp$ is K, on which we apply **Softmax** activation to retrieve $\mathbf{A}_t$.

Next, the model uses both $\mathbf{A}_t$ and $\mathbf{H}$ as inputs for the repetition layer, **Rep**. The repetition layer, **Rep**, outputs $\mathbf{R}$ of shape $K \times 2 * N_u$. $\mathbf{R}$ is simply the scalar multiplication of each hidden input $h_i \in \mathbf{H}$ with its corresponding scalar attention value, $a_i \in \mathbf{A}_t$.

Then, the model uses $\mathbf{R}$ as input for the aggregation layer, **Agg**. The aggregation layer simply computes the column-wise sum of $\mathbf{R}$ in order to yield the $2 * N_u$ shape output, $\mathbf{A}_g$. In short, $\mathbf{A}_g$ outputs the weighted sum of $\mathbf{H}$ where weights are the attention values.

$$\mathbf{A}_g = \mathbf{Summation}(\mathbf{R})$$

$\mathbf{A}_g$ is then used as input to a fully connected layer with sigmoid activation function, whose output is a scalar, $\hat{y}$, which denotes the prediction of a binary label, y.

$$\hat{y} = \mathbf{Sigmoid}(\mathbf{A_g} * \mathbf{W_3})$$

Here $\mathbf{W_3}$ is a randomly initialized weight matrix of shape $2 * N_u \times 1$.

Using these constructs, we train the binary classifier using the edge embeddings to predict whether the sentence S exhibits cause-effect relation for a given cause-effect pair. We train the model using standard binary cross-entropy loss: $Loss = -\frac{1}{|T|} \sum_{t \in T} y_t * log(\hat{y}_t) + (1 - y_t) * log(1 - \hat{y}_t)$. Using Early Stopping [31], we train the model until the validation loss does not decrease at the end of an epoch and then load the model parameters of the previous epoch in which validation loss has decreased.

While the model learns to classify, it identifies important edges based on attention values which contribute in classification. To make ASPER corpus independent, we introduce frequent itemset mining over the collected edges and extract the complete frequent dependency patterns. The statistics of the patterns and performance of ASPER can be found in our previous work [26].

3.2.3 Validation of Dependency Patterns and Template Creation

ASPER extracts cause-effect dependency pattern with high precision although partial and noisy patterns are extracted at times. Moreover, to ensure better recall for cause-effect pair extraction, we need to work with strong dependency patterns. Additionally, template patterns are easily convertible to dependency patterns and the scope of this paper is causality extraction from a sentence, which can be started from the pattern

templates as the templates are human recognizable patterns. The templates we provide can be modified if necessary for future work. Finally, filtering and validating extracted patterns are way easier than finding patterns from scratch. Below, we want to illustrate how we create templates from dependency pattern using an example.

Pattern template creation task is performed once dependency patterns are collected from ASPER. For an instance of dependency pattern, consider the example sentence in Fig. 1. In this sentence, the cause phrase *X* is *"bacterial or viral respiratory infections,"* and the effect phrase *Y* is *"Most AE-COPD cases"*. The following dependency pattern, ρ

$$\{(\textbf{attribute}, verb, nsubjpass, Y), (\textbf{attribute}, verb, prep, \textbf{to}), (\textbf{to}, adp, pobj, X)\}$$

is extracted by ASPER where *X* and *Y* can be any cause and effect term respectively. From this dependency pattern, we have introduced four pattern templates, *Y attributed to X*, *Y is attributed to X*, *Y can be attributed to X*, and *Y, which is attributed to X*. While there can be a lot more templates other than these, most of these template patterns lead to the same dependency pattern they come from. For example, if *Y is attributed to X* and *Y, which is attributed to X* are parsed with a dependency parser, and the dependency edges in the shortest path between *X* and *Y* are observed, the edges will be identical to the edges of ρ.

3.2.4 Regenerating Dependency Patterns from Pattern Templates

While most of the templates for a dependency pattern lead to the same dependency pattern, occasionally there can be some marginal changes in the dependency edges. Sometimes, POS tags are changed. For instance, for the sentence *Malaria is caused by the Plasmodium parasite*, *Malaria* is a *PROPN*. On the contrary, consider *Most fractures are caused by a bad fall or automobile accident*. Here the effect term is *Most fractures* which is *NOUN*. To overcome this, we generate a lot of dummy sentences using the pattern templates, replacing X and Y with some actual cause-effect terms, for example (*fire*, *damage*), (*Malaria*, *Plasmodium parasite*), etc. Once the sentences are constructed, we parse those sentences with dependency parser. Then, the shortest path edges are calculated with identical approach as ASPER described in Section 3.2.2. Actual cause-effect terms of those edges are replaced by *X* and *Y* for general dependency patterns. These final dependency patterns are stored in $\mathcal{P}_C$ for causality extraction. Column 1 of Fig. 4 shows some pattern templates, whereas column 3 shows the corresponding dependency patterns which are stored in $\mathcal{P}_C$. Note that, with the described method, we have 142 dependency patterns stored in $\mathcal{P}_C$, and all of the patterns of $\mathcal{P}_C$ are used by PATTERNCAUSALITY for causality extraction.

3.3 Pattern Matching and Extraction of Cause-Effect Candidates

For extracting cause and effect phrases from a test sentence, $\mathcal{L}$, we simply need to search whether the dependency tree of $\mathcal{L}$ has two nodes *u* and *v* such that the shortest

path between u and v matches with any pattern $\mathcal{P}$ in $\mathcal{P}_C$. Obviously, we do not know which pair of nodes may qualify the above test; we also do not know which pattern may appear in $\mathcal{L}$; so, we search over all possible pairs of nodes of $\mathcal{L}$'s dependency tree and all possible patterns. On some occasions, for a valid cause-effect relationship in $\mathcal{L}$, the match can be partial, i.e., only a subset of edges in the pattern may appear in the shortest path between u and v. To positively recall such cases, we consider a match to be acceptable only if a fraction (between 0.5 and 1) of pattern edges appear in the shortest path. This fraction is called *minThreshold* and it remains as a user-defined parameter. If the matching is successful, the words associated with u node are considered as cause phrase candidates and the words associated with v node are considered as effect phrase candidates. The experimental results that we show in this paper are generated using *minThreshold* equal to 1.

3.4 Complete Phrases Extraction from the Candidates

For many sentences in scientific literature, the candidate cause and effect phrases are not complete. So, we need to extend both the phrases, if such situation arises. To do that, we again take cues from the dependency tree. Assume that for a sentence $\mathcal{L}$, its dependency tree nodes u and v are found to be cause and effect candidates. Say, w_u and w_v are phrases of $\mathcal{L}$, which are associated to those two nodes, respectively. To extend the cause phrase we collect the ancestor and descendent nodes of u in the dependency tree. We then create a phrase w_a by concatenating the words associated with the ancestors of u by maintaining their order in $\mathcal{S}$. Likewise, all the successors of w_u are used to find another phrase w_s. Finally, between w_a and w_s, the phrase (let w_m) with the maximum number of words is considered as the extension of w_u. The extension, w_m, may contain part of pattern words, stop words, etc., so w_m is cleaned by removing such words to produce w_{cause}, which is our final cause phrase for the sentence $\mathcal{S}$. An identical process is applied for the node v to obtain the final effect phrase w_{effect}. The pair (w_{cause}, W_{effect}) is the extracted cause-effect phrase from the sentence $\mathcal{S}$. Note that a sentence S may have multiple cause-effect phrases for different pair of dependency tree nodes, in that case all such pairs are returned.

Please refer to Fig. 1 for a complete example of phrase extraction. The given sentence is: "Most AE-COPD cases are attributed to bacterial or viral respiratory infections and to both types of microorganisms together, which is represented with the dependency tree in Fig. 1. Consider the node u corresponding to *both types*, and v corresponding to *Most AE-COPD cases*. The shortest path between them is the following:

$$\{(attribute, verb, nsubjpass, v), (attribute, verb, prep, to),$$
$$(to, adp, conj, to), (to, adp, pobj, u)\} \tag{1}$$

A pattern exists in $\mathcal{P}_C$ with 100% match for the pair (*both types, Most AE-COPD cases*). Also, for the following pair $u =$*both types* and $v =$*bacterial or viral respiratory infection*, the shortest path between them is the following:

$$\{(attribute, verb, nsubjpass, v), (attribute, verb, prep, to), (to, adp, pobj, u)\} \quad (2)$$

Another pattern exists in $\mathcal{P}_C$ which shows 100% match with this pair. So, both of these pairs are considered as candidate cause-effect phrases. Now, the ancestors of *both types* are the singleton set {*to*}, as the other ancestors are not adjacent with *to* in the sentence $\mathcal{S}$. Successors of *both types* are the set {*of, microorganisms*}. After concatenation of the ancestors and successors with *both types*, we obtain, *to both types of microorganisms*; After cleaning up, the word "to" is omitted (because it is a stop word) and the final cause phrase becomes *both types of microorganisms*. The node *Most AE-COPD cases* and also the node *bacterial of viral respiratory infection* remains unchanged after the extension. Final results from this sentence are two cause-effect pairs: first is (*bacterial or viral respiratory infection, Most AE-COPD cases*), and the second is (*both types of microorganisms, Most AE-COPD cases*).

Algorithm 1 Cause-effect phrase extraction from sentence.

```
1  Extract-Causal-Phrase (S, P_C, minThreshold)
2      C = []
3      C_△ = []
4      T = DependencyParser(S)
5      for (u, v) ∈ S do
6          P = ShortestPath(T, u, v)
7          for P ∈ P_C do
8              mRatio = |P ∩ P| / |P|
9              if mRatio >= minThreshold then
10                 C_△.append((u, v))
11             end
12         end
13     end
14     for (m, n) ∈ C_△ do
15         C = Descendants(m)
16         u = MergeIfSequential(C)
17         u = Clean(u)
18         C = Descendants(n)
19         v = MergeIfSequential(C)
20         v = Clean(v)
21         C.append((u, v))
22     end
23     return C
24
```

We summarize the whole process with a pseudo-code in Algorithm 1. The method **Extract-Causal-Phrase** takes the sentence $\mathcal{S}$, the pattern collection $\mathcal{P}_C$, and the minimum threshold, *minThreshold* as parameters. Given the sentence, first we find the dependency tree of $\mathcal{S}$ (Line 4). Then in the nested for loop (Lines 5–12), for all node pairs u, v of $\mathcal{T}$ and for all patterns in $\mathcal{P}_C$, we obtain the shortest path between u and v, and check whether a significant fraction of the edges in the shortest path overlap with the pattern edges. If yes, the pair u and v is stored in C_Δ as the candidate cause-effect phrase pair. Then, for all phrase pairs in C_Δ, we extend and

clean them as needed (Lines 15–22). The overall complexity of the above method is quadratic with the number of words in a sentence and linear with the number of patterns.

4 Experimental Design

In this section, we show experimental results to validate the performance of our proposed method. For this, we use two datasets, SemEval and MedCause, which are discussed in detail in Section 4.1. Both the datasets contain sentences along with labeled cause and effect phrases and our objective is to extract these phrases from each of the sentences in unsupervised manner. The annotation is used only for evaluation. We compare the performance of PATTERNCAUSALITY with three competing methods, which are discussed in Section 4.2. We use precision, recall, and F_1 as evaluation metrics. In Section 4.3, we discuss how these metrics are computed in our experiments.

4.1 Datasets

SemEval   This is a well-used dataset, built by combining the SemEval 2007 Task 4 dataset [28] and the SemEval 2010 Task 8 dataset [29]. The SemEval datasets provide predefined positive and negative sentences with corresponding entity pairs. The datasets also include predefined train and test partitions. We made a validation partition by borrowing from train and test datasets through uniform sampling. In total, there are 7545 sentences in the train, out of those 922 are positive sentences. The validation dataset contains 166 positive sentences out of 1332 sentences in total. Lastly, the test dataset contains 3060 total number of sentences, out of which 339 are positive sentences. Among the cause-effect pairs, 90% are single words, 8% are double words, and the rest are multi-word nouns or noun phrases.

MedCause   We created this dataset by manual inspection of a dump of a large collection of PubMed articles. It contains 349 sentences, each of which are positive having confirmed cause-effect phrases. Three human experts labeled the cause and effect phrases from these sentences with 100% inter-expert agreement. Note that, each sentence can contain multiple cause-effect pairs, and thus can contribute more than one cause-effect pairs for the dataset. After extracting cause-effect pair, we have 446 rows in the dataset.

A key distinction of this dataset from SemEval is that in this dataset cause-effect phrases consist of relatively larger number of words. To demonstrate this, we draw a histogram showing the distribution of word count in the cause or effect phrase (we took the largest of these two count) as shown in Fig. 3. As can be seen the mode of the statistics is 5, and the majority of the phrases have a count of higher than 5. On the other hand, more than 90% of the cause or effect phrase in the SemEval dataset is single word, which makes the extraction of cause-effect phrases in the MedCause dataset much more difficult.

4.2 Competing Methods

To show the comparative performance of PATTERNCAUSALITY, we consider two competing methods. Out of these two methods, Logical-Rule Based method is a hybrid method, having both unsupervised and supervised components, and Word Vector Mapping Based method is purely unsupervised. More details of each of these methods are given below.

Logical-Rule Based This is a rule-based hybrid method proposed by Sorgente [18] where in the first step, a collection of cause-effect rules are used to extract cause-effect candidates in an unsupervised manner. Unlike the dependency patterns that we use in PATTERNCAUSALITY, these rules consist of different causative verbs in active or passive form, with or without preposition. These rules are matched in a given sentences to obtain cause and effect phrase candidates. However, not all the candidates they extract contain causal relationship. So, in a second step, they used a supervised binary classification to filter our false positive pairs. To train the classification model, they use the train partition of SemEval dataset, and classify based on that trained model.

For our experiment, we also do the same; when reporting results on the SemEval dataset using Logical-Rule based method, we build the supervised part of the model by using the train and the validation partition of the SemEval dataset and then report results on the corresponding test dataset. For experiments on the MedCause dataset, the same model is used. We also build another version of this classification model, which was trained with all SemEval train instances plus 20% of the random data instances from the MedCause dataset. Intention of building this second model is to validate whether retraining with sentences from the medical domain improves the performance of this method.

Fig. 3 Histogram for MEDCAUSE dataset, where X-axis shows maximum number of words in a causal term; Y-axis shows the frequency
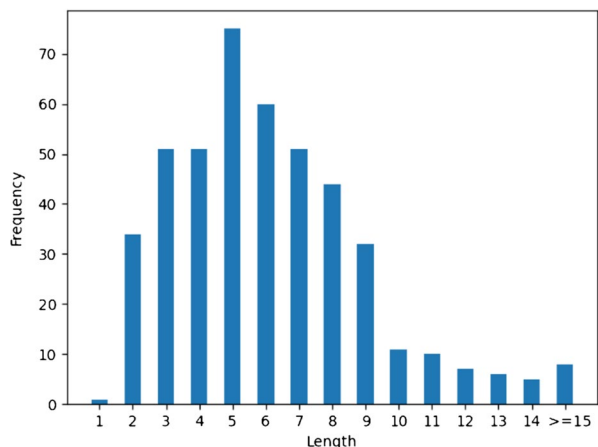
Word Vector Mapping Based This method is proposed for building a causal graph from a medical corpora, but this method can extract cause-effect terms as well [12]. It is an unsupervised method which uses regular expression-based dependency parsing. Then pre-trained Skip-Gram method of Word2Vec [32] is used to discover causative verbs with cosine similarity. From those causative verbs and regular expression based Parts of Speech parsing the authors extract cause-effect terms from sentences. Extracted cause-effect terms are then used to form a causal graph. We use the causality extraction ability of this method and introduce it as one of the competing methods.

Dependency Pattern–Based Variant of PATTERNCAUSALITY We create this baseline in which dependency patterns are used. So, in terms of sentence representation, this method is identical to PATTERNCAUSALITY. However, for extracting phrases, this method relies on the noun phrase extraction [33, 34] methods whose implementation is available from Spacy [22]. A problem with Spacy's noun phrase extraction processes is that they often extract only a single word, instead of complete cause and effect phrases. In order to enhance the phrase extraction performance, we also use an advanced phrase extraction technique, namely PKE [35].

Note that, PKE can be used with earlier baseline methods also to improve their phrase extraction capability, so we present results for a second variant of all the competing methods in which PKE is used for extracting phrases. So, in total, we have 8 methods for which we show comparison results: 3 logical-rule–based methods, 2 word vector mapping–based method, and finally PATTERNCAUSALITY and two of its dependency tree–based variants.

Besides the above methods, there are some supervised methods available in the literature which can extract causality [15, 16]; however, we do not consider these methods for comparison. This is due to the fact that their performance is highly dependent on the datasets on which they are trained on. For our task of extracting cause-effect phrases from scientific literature or medical domain, there are no annotated corpus available for supervised training, a fact corroborated by other researchers [12]. We do have the options of training these models using SemEval datasets, which we tried; but such a trained model performs very poorly on Med-Cause sentences due to highly different distribution data between MEDCAUSE and SemEval corpus. Being the fact that our method is unsupervised, we limited our comparison to the above three methods, for which the phrase extraction part is unsupervised.

4.3 Evaluation Metrics

For evaluation, we use traditional binary classification metrics, such as precision, recall, and F_1-score over the sentences. However, as complete phrase extraction is a difficult task, we define Levenshtein similarity–based evaluation metrics which allow some margins of freedom to each method in terms of phrase extraction. In

this subsection, we want to define all the evaluation metrics, and performance of the methods will be discussed in the later subsection.

For a sentence $\mathcal{S}$, let (x, y) be a cause-effect term pair predicted by any of the described methods. As there can be multiple cause-effect pairs per sentence, we define $\mathcal{C}_p$ as a set containing all cause-effect-sentence triplets, $(x, y, \mathcal{S})$. Similarly, let $\mathcal{C}_t$ be a set for test triplets. Overall precision and recall are then defined by the following equations.

$$Prec = \frac{|\mathcal{C}_p \cap \mathcal{C}_t|}{|\mathcal{C}_p|} \tag{3}$$

$$Rec = \frac{|\mathcal{C}_p \cap \mathcal{C}_t|}{|\mathcal{C}_t|} \tag{4}$$

Extracting a phrase from a sentence is not an absolute task, it depends on the perspective of a viewer. So, we have introduced another measure of precision and recall with Levenshtein similarity ratio which we call edit similarity ratio. Let two phrases be s_1 and s_2. If only insertion, deletion, or substitution of a character is allowed to convert one phrase to another, let λ be a minimum number of such operations needed for this conversion. If $|s_1|$ and $|s_2|$ are total number of characters in s_1 and s_2, respectively, then Levenshtein similarity ratio, *ePer* is defined by (5). *ePer* similarity gives a value between 0 and 100, so we use a threshold parameter, *minSim*, on the similarity value which denotes the minimum similarity necessary for two phrases for being similar. Thus when comparing a predicted cause-effect phrase with a ground truth cause-effect phrase, if *ePer* >= *minSim*, we count the prediction as correct.

$$ePer = \frac{max(|s_1|, |s_2|) - \lambda}{max(|s_1|, |s_2|)} \times 100\% \tag{5}$$

5 Results

In this section, we want to provide an extensive evaluation of PATTERNCAUSALITY on two benchmark datasets described before. We report precision, recall, and F_1 scores for both evaluation types (with and without *minSim*; without *minSim* means *minSim* threshold is set to 100%). We also present our results by grouping the phrases based on the number of words in the phrases to compare each method's ability of extracting longer medical phrases. However, such results are provided only for MEDCAUSE dataset as 90% of cause or effect phrases of SemEval datasets are single word, as noted in Section 4.1.

5.1 Comparison with Competing Methods on MEDCAUSE Dataset

First, we present comparison results of different cause-effect phrase extraction methods on the MEDCAUSE dataset using precision, recall, and F_1 metrics. Table 1 shows

Table 1 Comparison with baseline methods on MEDCAUSE dataset when *minSim* = 100%

Method	Prec	Rec	F_1
Logical-Rule Based [11]	0.020	0.038	0.026
Logical-Rule Based + PKE	0.021	0.038	0.026
Enhanced Logical-Rule Based + PKE	0.024	0.039	0.030
Word Vector Mapping Based [12]	0.028	0.041	0.033
Word Vector Mapping Based + PKE	0.031	0.049	0.038
Dependency Pattern Based	0.142	0.135	0.138
Dependency Pattern Based + PKE	0.512	0.428	0.520
PatternCausality	**0.556**	**0.530**	**0.543**

Table 2 Comparison with baseline methods on MEDCAUSE dataset when *minSim* = 80%

Method	Prec	Rec	F_1
Logical-Rule Based [11]	0.044	0.110	0.063
Logical-Rule Based + PKE	0.044	0.112	0.063
Enhanced Logical-Rule Based + PKE	0.048	0.118	0.068
Word Vector Mapping Based	0.055	0.148	0.080
Word Vector Mapping Based + PKE	0.062	0.152	0.088
Dependency Pattern Based	0.215	0.211	0.212
Dependency Pattern Based + PKE	0.592	0.513	0.560
PatternCausality	**0.609**	**0.589**	**0.600**

the results considering exact match (*minSim* = 100%), whereas Table 2 shows the results for 80% or more edit similarity (*minSim* = 80%). As we can see a total of eight methods are shown, which are grouped (groups are separated by horizontal line) based on their methodologies. Our proposed method and its variants are in the last group.

Among all the methods, PATTERNCAUSALITY and its variants, which use syntactic dependency patterns perform substantially better than both Logical-Rule Based and Word Vector Mapping Based methods. As we can see in the tables, for exact match and partial match, PATTERNCAUSALITY's F_1 values are respectively, 0.543 and 0.600, whereas the best values for the same among the competing groups are 0.038 and 0.088. Clearly, PATTERNCAUSALITY's performance is at least one order of magnitude better than the best of the methods in the competing groups. In fact, the performance of other dependency pattern–based methods that we have proposed as baselines, though worse than PATTERNCAUSALITY, is better than Logical-rule based or Word vector mapping based methods by nearly one order of magnitude. For example, Dependency Pattern based + PKE has F_1 values of 0.520 and 0.560, which is the second best result overall after PATTERNCAUSALITY. These results illustrate that dependency pattern–based approach which we propose is much superior than the existing approach for extracting cause-effect phrases.

All of the dependency pattern–based approaches recognize the cause and effect nodes in the dependency tree using patterns. But, they differ in the way they extract

the phrases. One of the baselines, Dependency pattern + PKE method uses PKE (phrase keyword extraction) [35] for phrase extraction. On the other hand, PATTERN-CAUSALITY uses a custom phrase extraction process for capturing longer phrases, which makes it better than other dependency pattern based methods. In summary, two-fold contribution of PATTERNCAUSALITY, first using dependency patterns for identifying cause and phrase nodes, then innovative phrase extraction makes it the winner among all the methods that we have shown in these tables.

The poorest performers among all the methods are Logical-Rule based methods. In fact, logical rules cannot identify cause-effect terms well; this is because the rules to extract causal terms are not adequate, and the rules are mainly logical rules which are not aware of syntactic structure of a sentence. Even if we apply PKE for phrase extraction, such a method still suffers. Note that, in the basic Logical-rule based method (Row 1), rules are obtained from the SemEval dataset; so in the Enhanced Logical-rule based method + PKE (Row 3), we borrowed 20% data from MEDCAUSE to SemEval in anticipation of getting better rules, yet the performance hardly improved.

Word vector mapping–based methods performs better than logical-rule–based methods. Such methods, although finding the causative verbs with cosine similarity, fails to capture the syntactic structure of a sentence. Moreover, not all the causative verbs are equally effective for exhibiting cause-effect relation. Another reason for poor performance for this method is that it does not have any specific phrase extraction technique which is needed for MEDCAUSE dataset where cause-effect phrases are relatively longer. We tried to enhance this method with PKE-based phrase extraction, which improved its performance noticeably, as can be seen from Tables 1 and 2; yet, the improved performance is substantially poorer than PATTERNCAUSALITY and its other variants.

5.2 Comparison with Competing Methods on SemEval Dataset

In Table 3, we show results of our methods and other competing methods on SemEval dataset for exact match scenario. As most of the cause-effect terms are single-word nouns in this dataset, all methods perform much better on this dataset and their performance is somewhat similar. The best performance is shown by dependency pattern–based method, one of PATTERNCAUSALITY variant. Its F_1-score is 0.61, whereas best among word vector–based and logical-rule–based method is 0.58 and 0.57, respectively. This validates that dependency pattern that we propose in PATTERNCAUSALITY is the best tool for extracting cause and effect phrase even for single-word scenarios. Interestingly, in all these methods, using dedicated phrase extraction tools, like PKE or the one that PATTERNCAUSALITY uses, makes the result worse. This is due to the fact that PKE or other phrase extraction method tries to make the cause and effect phrase longer, but mostly all the cause-effect phrases in this dataset are of single-word length.

5.3 Performance of the Competing Methods Grouped by Phrase Length

The key contribution of our method is that it can extract longer cause-effect phrases, whereas existing methods fail to do so. In this experiment, we validate that the

Table 3 Performance of the methods in SemEval dataset when minSim = 100%

Method	Prec	Rec	F_1
Logical-Rule Based	0.71	0.47	0.57
Logical-Rule Based + PKE	0.63	0.39	0.48
Word Vector Mapping Based	0.74	0.47	0.58
Word Vector Mapping Based + PKE	0.69	0.45	0.54
Dependency Pattern Based	**0.75**	**0.51**	**0.61**
Dependency Pattern Based + PKE	0.7	0.47	0.56
PATTERNCAUSALITY	0.71	0.48	0.57

performance of existing methods increasingly becomes worse as the length of the phrase increases. In Fig. 3, we have shown that the length of the majority of the causal terms in MEDCAUSE dataset is between 2 and 9. So, we partition the MEDCAUSE test dataset based on the phrase length and then show the performance of each method on each of those partitions in Table 4. We can see that over all different lengths, PAT-TERNCAUSALITY has good performance. But for the competing methods, their performance drops significantly as the length increases. For several of the competing methods, the performance drops to 0 when the length of the phrases reaches 5 or more.

5.4 Performance of Selected Dependency Patterns

PATTERNCAUSALITY's main contribution is to use dependency patterns to extract cause and effect phrases. To demonstrate the role of dependency patterns, in Fig. 4, we show a selected set of dependency patterns along with an example sentence from the MEDCAUSE dataset, and its associated cause and effect phrases. We also perform experiments to how precise a pattern is, i.e., how well a pattern can extract the cause and effect phrases after it has been successfully used in a sentences. So, we define a metric, named pattern precision, which defines the ratio of the number of correctly predicted phrases over the total number of phrases predicted by a pattern.

Table 4 Performance of all the methods in MEDCAUSE dataset for most densely populated lengths

Method	Length of cause of effect phrase							
	2	3	4	5	6	7	8	9
Word Vector Mapping Based	0.317	0.161	0.001	0	0	0	0	0
Word Vector Mapping + PKE	0.330	0.202	0.021	0.013	0.016	0	0	0
Logical-Rule Based	0.281	0.125	0.001	0	0	0	0	0
Logical-Rule Based + PKE	0.312	0.140	0.023	0.015	0.024	0	0	0
Enhanced Logical-Rule Based + PKE	0.310	0.160	0.031	0.025	0.031	0	0	0
Dependency Pattern Based	0.482	0.173	0.031	0	0	0	0	0
Dependency Pattern + PKE	0.481	0.193	0.062	0.030	0.022	0.010	0	0
PatternCausality	**0.562**	**0.557**	**0.543**	**0.463**	**0.435**	**0.422**	**0.417**	**0.414**

Precision of total 12 selected patterns with exact match and 80% match is shown in Fig. 4. Obviously, the precision values are higher for the case of partial (80%) match than exact match scenario. The performance of rules is mixed: exact match precision varying between .50 and 1.00. Note that, rules only identify the dependency tree nodes associated to head words of cause and effect phrases, from where complete phrases are extracted by using the phrase extraction method. So, precision of a rule is also affected by the subsequent phrase extraction process. Most of the rules that we have used have precision better than 0.5.

6 A Case Study

Finally, we demonstrate the ability of PATTERNCAUSALITY to extract full cause and effect phrases by showing an example sentence from the MEDCAUSE dataset and analyzing how different methods perform on this sentence. We select the following

Cause-Effect Pattern	Example Cause-Effect Pair	Cause-Effect Dependency Pattern	Precision	Precision with minSim = 80%
X caused Y	SM21, reduction of 12% in cell viability	SM21 caused a reduction of 12% in cell viability.	0.738	0.872
X precipitated Y	Gastrointestinal losses, hypokalemia	Gastrointestinal losses and anorexia can precipitate hypokalemia	1.000	1.000
X triggered Y	a variety of physiological and pathological stimuli, apoptosis	A variety of physiological and pathological stimuli may trigger apoptosis.	0.500	0.833
X associated with Y	SNPs, a poor prognosis from sepsis	SNPs may be associated with a poor prognosis from sepsis.	0.522	0.627
Y caused by X	mucins, T elevation	Neither Infasurf nor SRB reduces the T elevation caused by mucins.	0.562	0.733
Y produced by X	amplification of the plasmid, the vector	The vector was produced by amplification of the plasmid.	0.625	0.824
Y triggered by X	loss or alteration of cell-cell or cellmatrix anchorage, a form of apoptosis	A form of apoptosis triggered by loss or alteration of cell-cell or cellmatrix anchorage ...	0.667	0.714
Y induced by X	DpgdA, the inflammatory response	The inflammatory response induced by DpgdA is due to . . .	0.750	1.000
Y attributed to X	cuts in the financing of public health, the lack of hospital beds	The lack of hospital beds has been attributed to cuts in the financing of public health.	0.714	0.857
Y resulted from X	acute ozone exposure, oxidative stress	Many of the changes are likely due to oxidative stress resulting from acute ozone exposure.	0.545	0.636
Y is due to X	congenital defects, organ failure	In many cases, organ failure is due to congenital defects.	0.545	0.727
Y is the result of X	a direct effect of the virus on CNS cells, abnormal neurologic functions	Abnormal neurologic functions are the result of a direct effect of the virus on CNS cells.	0.500	0.750

Fig. 4 Cause-effect pairs extraction result for some selected dependency patterns

sentence: "Moreover, amino acid sequence mutations in the new variant strains will cause immunization failure of commercial vaccines." The causal phrase is "amino acid sequence mutations in the new variant strains", and the effect phrase is "immunization failure of commercial vaccines". PATTERNCAUSALITY extracts both the cause and effect phrases exactly. Logical-Rule Based and other baseline approaches extract "the new variant strains" as the cause phrase, and "immunization failure" as an effect phrase—incomplete phrases for both cause and effect. Dependency Pattern Based approach and Dependency Pattern + PKE extract "amino acid sequence mutations" and "immunization failure" as cause and effect terms respectively, which are also incomplete.

7 Conclusion and Future Work

In medical domain, causality extraction from literature is a very important task for knowledge extraction, literature-based review, and hypotheses generation. But, existing cause-effect phrase extraction methods are highly inadequate for solving this task with high accuracy. In most of the cases, with existing methods, the extracted causality phrases are incomplete, which leads to knowledge that to the best, is confusing, and to the worst, is inaccurate. Since, no existing methods pursue this task specifically for medical domain, we first created a manually annotated dataset, MEDCAUSE, which is the first dataset of its kind. This is an important contribution towards the medical information retrieval domain. Then we have contributed a novel method, PATTERN-CAUSALITY, for causality extraction. Our proposed method is unsupervised, so it does not need large annotated corpus for training, which makes it immediately usable. It is also extendable, as more dependency patterns can be added to the pattern library to improve its performance. Finally, we have demonstrate through detailed experiments that PATTERNCAUSALITY is highly effective to extract long cause and effect phrases, whereas other competing methods fail to do so. We also build other variants of PATTERNCAUSALITY, which uses only dependency patterns or uses a different phrase extraction tool, namely PKE to demonstrate that dependency pattern–based cause-effect phrase extraction is an effective unsupervised approach.

In this work, we do not compare PATTERNCAUSALITY with any supervised approach, such as LSTM or other sequence-based model. The reason for that is lack of large datasets for the purpose of training. So, one of the future goals is to first extract and annotate adequate sentences using PATTERNCAUSALITY and then train an effective supervised model for solving this task. Secondly, we have observed that the words considered for larger phrases are sequential in the Spacy dependency tree for majority cases. However, there are exceptions in this assumption. We want to extend PATTERNCAUSALITY in future to deal with those cases. Authors are committed to reproducible research and they will publish the MEDCAUSE dataset, code, and the dependency patterns, once this paper is accepted.

Declarations

Conflict of Interest The authors declare no competing interests.

References

1. Wald D, Law M, Morris J (2002) Homocysteine and cardiovascular disease: Evidence on causality from a meta-analysis. BMJ (Clinical research ed.) 325:1202. https://doi.org/10.1136/bmj.325.7374.1202

2. Azagi T, Sprong H, Hoornstra D, Hovius J (2020) Evaluation of disease causality of rare ixodes ricinus-borne infections in europe. Pathogens 9:150. https://doi.org/10.3390/pathogens9020150

3. Martín Giménez V, Inserra F, Ferder L, García J, Manucha W (2020) Vitamin D deficiency in african americans is associated with a high risk of severe disease and mortality by Sars-CoV-2. (Letter) https://doi.org/10.1038/s41371-020-00398-z

4. Evans W, McLeod H (2003) Pharmacogenomics-drug disposition, drug targets, and side effects. The New England J Med 348:538–49. https://doi.org/10.1056/NEJMra020526

5. Berry D, Knapp P, Raynor D (2002) Provision of information about drug side-effects to patients. Lancet 359:853–4. https://doi.org/10.1016/S0140-6736(02)07923-0

6. Dicpinigaitis P (2006) Angiotensin-converting enzyme inhibitor-induced cough: ACCP evidence-based clinical practice guidelines. Chest. 129:169S-173S. https://doi.org/10.1378/chest.129.1_suppl.169S

7. Atkinson J, Rivas A (2008) Discovering novel causal patterns from biomedical natural-language texts using bayesian nets. IEEE Trans Inf Technol Biomed 12(6):714

8. Lee DG, Shin H (2017) Disease causality extraction based on lexical semantics and document-clause frequency from biomedical literature. BMC Medical Inform Decis Mak 17(1):53

9. Zhao S, Jiang M, Liu M, Qin B, Liu T (2018) Causaltriad: toward pseudo causal relation discovery and hypotheses generation from medical text data. In: Proceedings of the 2018 ACM international conference on bioinformatics, computational biology, and health informatics, pp 184–193

10. An N, Xiao Y, Yuan J, Yang J, Alterovitz G (2019) Extracting causal relations from the literature with word vector mapping. Comput Biol Med 115:103524

11. Sorgente A, Vettigli G, Mele F (2013) Automatic extraction of cause-effect relations in natural language text. DART@ AI* IA 2013,37

12. An N, Xiao Y, Yuan J, Jiaoyun Y, Alterovitz G (2019) Extracting causal relations from the literature with word vector mapping. Comput Biology Med 115:103524. https://doi.org/10.1016/j.compbiomed.2019.103524

13. Khoo C, Kornfilt J, Oddy R, Myaeng SH (1998) Automatic extraction of cause-effect information from newspaper text without knowledge-based inferencing. Literary and Linguistic Computing 13:177–186. https://doi.org/10.1093/llc/13.4.177

14. Khoo C, Myaeng SH, Oddy R (2001) Using cause-effect relations in text to improve information retrieval precision. Inf Process Manage 37:119–145. https://doi.org/10.1016/S0306-4573(00)00022-4

15. Dasgupta T, Saha R, Dey L, Naskar A, Automatic extraction of causal relations from text using linguistically informed deep neural networks. In: Proceedings of the 19th annual sigdial meeting on discourse and dialogue (Association for Computational Linguistics, Melbourne, Australia, 2018), pp. 306–316. https://doi.org/10.18653/v1/W18-5035

16. Li Z, Li Q, Zou X, Ren J (2019) Causality extraction based on self-attentive BiLSTM-CRF with transferred embeddings

17. Chang DS, Choi KS (2006) Incremental cue phrase learning and bootstrapping method for causality extraction using cue phrase and word pair probabilities. Neurocomputing 42:662–678. https://doi.org/10.1016/j.ipm.2005.04.004

18. Sorgente A, Vettigli G, Mele F (2013) Automatic extraction of cause-effect relations in natural language text. 1109:37–48

19. Do Q, Chan YS, Roth D (2011) Minimally supervised event causality identification. In: Proceedings of the 2011 conference on empirical methods in natural language processing, pp 294–303

20. Nguyen TH, Grishman R (2015) Relation extraction: Perspective from convolutional neural networks. In: Proceedings of the 1st workshop on vector space modeling for natural language processing, pp 39–48

21. Peng N, Poon H, Quirk C, Toutanova K, Yih WT (2017) Cross-sentence n-ary relation extraction with graph lstms. Trans Assoc Comput Linguist 5:101

22. Honnibal M, Montani I (2020) spaCy 2.2.3: Industrial-strength natural language processing. In: https://spacy.io/

23. Marneffe MC, Manning C (2008). The Stanford typed dependencies representation. https://doi.org/10.3115/1608858.1608859

24. Choi JD, Palmer M (2012) Fast and robust part-of-speech tagging using dynamic model selection. In: Proceedings of the 50th annual meeting of the association for computational linguistics (volume 2: short papers) (Association for Computational Linguistics, Jeju Island, Korea), pp 363–367. https://aclanthology.org/P12-2071

25. Johansson R, Nugues P (2007) Extended constituent-to-dependency conversion for english. In: Proceedings of the 16th nordic conference of computational linguistics (NODALIDA 2007) (University of Tartu, Estonia, Tartu, Estonia, 2007), pp 105–112. https://aclanthology.org/W07-2416

26. Kabir MA, Phillips T, Luo X, Hasan MA (2021) Asper: Attention-based approach to extract syntactic patterns denoting semantic relations in sentential context

27. Gurulingappa H, Mateen A, Roberts A, Fluck J, Hofmann-Apitius M, Toldo L (2012) Development of a benchmark corpus to support the automatic extraction of drug related adverse effects from medical case reports. J Biomed Inf. https://doi.org/10.1016/j.jbi.2012.04.008

28. Girju R, Nakov P, Nastase V, Szpakowicz S, Turney P, Yuret D (2007) SemEval-2007 task 04: classification of semantic relations between nominals. 13–18

29. Hendrickx I, Kim S, Kozareva Z, Nakov P, Padó S, Pennacchiotti M, Romano L, Szpakowicz S (2010) SemEval-2010 task 8: Multi-way classification of semantic relations between pairs of nominals. 33–38

30. Zaki MJ (2000) Scalable algorithms for association mining. IEEE Trans Know Data Eng 12(3):372. https://doi.org/10.1109/69.846291

31. Caruana R, Lawrence S, Giles CL (2001) Overfitting in neural nets: backpropagation, conjugate gradient, and early stopping. In: Leen TK, Dietterich TG, Tresp V (eds) (Advances in neural information processing systems 13, MIT Press, 2001), pp 402–408. http://papers.nips.cc/paper/1895-overfitting-in-neural-nets-backpropagation-conjugate-gradient-and-early-stopping.pdf

32. Mikolov T, Corrado G, Chen K, Dean J (2013) Efficient estimation of word representations in vector space. 1–12

33. Handler A, Denny M, Wallach H, O'Connor B (2016) Bag of what? simple noun phrase extraction for text analysis. 114–124. https://doi.org/10.18653/v1/W16-5615

34. Subhashini R, Kumar VJS (2010) Shallow NLP techniques for noun phrase extraction. In: Trendz in information sciences computing(TISC2010), pp 73–77. https://doi.org/10.1109/TISC.2010.5714612

35. Boudin F (2016) pke: an open source python-based keyphrase extraction toolkit. In: Proceedings of COLING 2016, the 26th international conference on computational linguistics: system demonstrations (Osaka, Japan, 2016), pp 69–73. http://aclweb.org/anthology/C16-2015