LearnIt: On-Demand Rapid Customization for Event-Event Relation Extraction

Bonan Min, Manaj Srivastava, Haoling Qiu, Prasannakumar Muthukumar, Joshua Fasching
Raytheon BBN Technologies, Cambridge, Massachusetts 02138

Abstract

We present a system which allows a user to create event-event relation extractors on-demand with a small amount of effort. The system provides a suite of algorithms, flexible workflows, and a user interface (UI), to allow rapid customization of event-event relation extractors for new types and domains of interest. Experiments show that it enables users to create extractors for 6 types of causal and temporal relations, with less than 20 minutes of effort per type. Our system (source code, UI) is available at https://github.com/BBN-E/LearnIt. A demonstration video is available at https://vimeo.com/329950144.

Extracting relations (e.g., a *flood* caused *migration*) between real-world events from natural language text is very useful for situation awareness and decision making. However, creating an event-event relation extractor often requires a significant amount of time and effort. For example, a developer may need to write a large set of extraction rules by hand, or curate a large labeled data set to train a classifier. Such approach will not be applicable for new relation types nor new genres of text different from the training data.

We developed LearnIt [1], a system for on-demand rapid customization of event-event relation extractors with a user in the loop. It has the following key features:

First, it incorporates **bootstrapping** to iteratively learn event pairs from patterns, and patterns from pairs, by leveraging an unannotated development corpus.

Second, it incorporates **iterative expansion** of its relational pattern/event-pair set through iteratively adding patterns/pairs that are similar to the set.

Third, it involves a **human in the loop** to prevent semantic drift in the bootstrapping and iterative expansion processes. We develop a UI to allow the user to review/select examples and steer the customization process, with a small amount of effort.

Related Work

There are several systems, e.g., (He and Grishman 2015; Li et al. 2012; Gupta and Manning 2014) for rapid customization for entities or relations between entities. None of them is designed for event-event relation extraction.

System Description

LearnIt aims at learning patterns that can be applied to text for extracting event-event relations. A pattern is 1) a lexical pattern, which is a sequence of words between a pair of events, e.g., "*0 leads to 1*" [2], or 2) a proposition pattern, which is the (possibly nested) predicate-argument structure that connects the pair of events. For example, "*verb:cause[subject=0] [object=1]*" is the proposition pattern counterpart of "*0 causes 1*".

For development purpose, LearnIt uses a large, unannotated corpus, processed by SERIF NLP toolkit (Boschee, Weischedel, and Zamanian 2005) to generate propositions and events. Following Richer Event Description [3], we adopted a broad definition for event: an event can be any occurrence, action, process or event state. Therefore, we tag all predicate-like verbs and nominalizations as event triggers.

As shown in Figure 1, the LearnIt system incorporates two workflows, bootstrapping and iterative pattern/pair set expansion, in its iterative learning process. Learnit also allows flexible compositions of these two workflows, mediated by the user, to allow more effective use of users' effort. The learning process will be guided with a small amount of user effort provided via a UI [4].

Workflow 1: Bootstrapping: LearnIt incorporates bootstrapping (Agichtein and Gravano 2000) for relation extraction. It works as follows: Given a handful of initial event pairs that are known to express the target relation, LearnIt searches in a development corpus to find instances (sentences mentioning the event pair). From these instances, LearnIt extracts relational patterns, ranks and presents them to the user. The user then selects patterns that express the target relation. These patterns are added into the known pattern set. Similarly, given a set of known patterns, LearnIt again searches in the corpus to find matched instances, from which

[1] **Learn**ing extractors **It**eratively.

[2] The left and right arguments of an relation are numbered 0 and 1 respectively. We focus on binary relations.

[3] https://github.com/timjogorman/RicherEventDescription

[4] LearnIt's web-based UI is shown in the demo video.

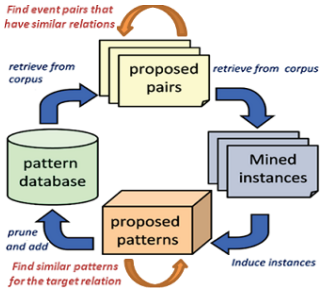

Figure 1: LearnIt workflows. Bootstrap learning is illustrated with the blue arrows and text. Iterative self-expansion is illustrated in orange self loops and the orange text.

it extracts additional event pairs, ranks and presents them to the user. The user selects event pairs that express the target relation. The user can perform multiple iterations of bootstrapping. An complete iteration is illustrated in Figure 1.

Workflow 2: Pattern/pair set expansion: This workflow incorporates ideas from distributional-similarity-based paraphrase and entity set expansion. Given a set of seed patterns expressing the target relation, LearnIt ranks all other patterns based on their similarity to the known patterns and presents a ranked list to the user. The user adds good patterns into the known pattern list. The process repeats iteratively. This allows the user to iteratively expand the list of patterns for the target relation with a small amount of effort. Similarly, it also allows the user to select additional event pairs that indicate the target relation, if event pairs are provided as seeds. This workflow is illustrated as orange loops in Figure 1.

To learn a continuous vector representation ("embeddings") of patterns and events for pairwise similarity calculation, we train a joint text and relation embedding algorithm (Toutanova et al. 2015) using 13 million $<\text{event}_1$, pattern, $\text{event}_2>$ triples generated from English Gigaword [5].

Given a set of known patterns, we rank all other patterns according to cosine similarity of their embeddings to the average embedding of the known pattern set. Similarly, given a set of known event pairs, we rank all other event pairs according to the cosine similarity of their embeddings to average embedding of the known pairs. The embedding of an event pair is the concatenation of the two event embeddings.

Experiments

Dataset We randomly sampled 1.5 million documents from English Gigaword as our development corpus, and sampled another 500 documents as the test corpus. For the test corpus, we ask annotators to annotate 6 types of temporal and causal relations [6] exhaustively for pairs of events appearing in the same sentence. The relations are defined in Table 1. The final annotation dataset contains 629 positive instances.

We asked users to use LearnIt to build relation extractors for the 6 relations from scratch, using the development corpus. On average, a user spent 18.7 minutes per type and found 134 patterns per type. These patterns were applied to the test corpus to extract relation instances. The relation extractors achieved good performance (Table 2).

[5] https://catalog.ldc.upenn.edu/LDC2011T07

[6] A relation is annotated if a trigger word is present. This is necessary in order to achieve a high inter-annotator agreement.

Type	Definition
Cause	Y happens because of X.
Preventative	If X happens, Y can't happen.
Precondition	X must have occured for Y to happen.
Catalyst	If X, intensity of Y increases.
Mitigation	If X, intensity of Y decreases.
Occurs before	X happens before Y.

Table 1: Definitions of relations (between event X and Y).

Relation type	Precision	Recall	F1
Cause	0.88	0.63	0.73
Preventative	0.67	0.52	0.59
Precondition	0.69	0.74	0.71
Catalyst	0.88	0.37	0.52
Mitigation	0.55	0.34	0.42
Occurs before	0.70	0.66	0.68

Table 2: Performance of LearnIt relation extractors.

Acknowledgments

This work was supported by DARPA/I2O and U.S. Army Research Office Contract No. W911NF-18-C-0003 under the World Modelers program. The views, opinions, and/or findings contained in this article are those of the author and should not be interpreted as representing the official views or policies, either expressed or implied, of the Department of Defense or the U.S. Government. This document does not contain technology or technical data controlled under either the U.S. International Traffic in Arms Regulations or the U.S. Export Administration Regulations.

References

Agichtein, E., and Gravano, L. 2000. Snowball: Extracting relations from large plain-text collections. In *Proceedings of the fifth ACM conference on Digital libraries*, 85–94. ACM.

Boschee, E.; Weischedel, R.; and Zamanian, A. 2005. Automatic information extraction. In *Proceedings of the International Conference on Intelligence Analysis*.

Gupta, S., and Manning, C. 2014. Spied: Stanford pattern based information extraction and diagnostics. In *Proceedings of the Workshop on Interactive Language Learning, Visualization, and Interfaces*, 38–44.

He, Y., and Grishman, R. 2015. Ice: Rapid information extraction customization for nlp novices. In *Proceedings of NAACL System Demonstrations*, 31–35.

Li, Y.; Chiticariu, L.; Yang, H.; Reiss, F. R.; and Carreno-Fuentes, A. 2012. Wizie: a best practices guided development environment for information extraction. In *Proceedings of the ACL 2012 System Demonstrations*.

Toutanova, K.; Chen, D.; Pantel, P.; Poon, H.; Choudhury, P.; and Gamon, M. 2015. Representing text for joint embedding of text and knowledge bases. In *Proceedings of EMNLP*.