Weak Supervision using Linguistic Knowledge for Information Extraction

Sachin Pawar[1], Girish K. Palshikar[1], Ankita Jain[1], Jyoti Bhat[1] and Simi Johnson[2]
[1]TCS Research, Tata Consultancy Services, Pune, India.
[2]Delivery Governance, Risk & Security, Tata Consultancy Services, Chennai, India.
`{sachin7.p,gk.palshikar,ankita7.j,jyoti.bhat1,simi.johnson}@tcs.com`

Abstract

In this paper, we propose to use linguistic knowledge to automatically augment a small manually annotated corpus to obtain a large annotated corpus for training Information Extraction models. We propose a powerful patterns specification language for specifying linguistic rules for entity extraction. We define an *Enriched Text Format* (ETF) to represent rich linguistic information about a text in the form of XML-like tags. The patterns in our patterns specification language are then matched on the ETF text rather than raw text to extract various entity mentions. We demonstrate how an entity extraction system can be quickly built for a domain-specific entity type for which there are no readily available annotated datasets.

1 Introduction

Much knowledge in an organization resides in text documents of various types. Effectively using the information and knowledge hidden in enterprise document repositories is a challenge. *Information Extraction (IE)* is a well-explored language processing technology for extracting specific kinds of information (e.g., generic or domain-specific entities, relations among entities and events) from documents and presenting it in a structured format (Palshikar, 2012; Pawar et al., 2017; Li et al., 2020). This structured information can then be effectively searched, disseminated, reused or mined using data mining techniques to discover valuable knowledge. IE plays a critical role in several applications such as resumes processing, competitor intelligence from news, patent analysis, and insurance claim management.

IE is posed as a classification task in machine learning, where the training data consists of labeled mentions of a given entity (or relation, or event) type in sentences. Creating a sufficient quantity of such training data is time-consuming and error-prone. Hence, there has been research in distant supervision methods for automating the process of creating training data, often using other knowledge sources such as DBPedia. In this paper, we map each sentence in the given corpus to an *enriched text format (ETF)* by adding syntactic and semantic information to raw text. We then propose an enriched regular expression language to write linguistic knowledge (rules) to extract mentions of entities and relations from the ETF representation of the sentences. Unlike tools like Snorkel (Ratner et al., 2017) or more complex tree regex languages, our pattern language is simpler, more efficient and has novel features to allow linguistic patterns that use context from multiple sentences. We demonstrate the use of linguistic knowledge to automatically create training data and show that the new training data improves accuracy of IE classification models. We demonstrate this methodology on a completely novel application that extracts risk factors from audit reports of software projects.

2 Linguistic Rules

We express linguistic rules in the form of regular expression patterns which are applied on an ETF text in which rich linguistic information is embedded in the form of XML-like tags.

2.1 Enriched Text Format

We use spaCy (Honnibal and Montani, 2017) for processing any input text and the ETF text is generated as follows: (see examples in Table 1)
• `<SENT>` and `</SENT>` tags are added to mark beginning and end of each sentence.
• Each token identified by the word tokenizer is then encapsulated by its corresponding part-of-speech tag. E.g., `<NN>connection </NN>`.
• Each generic named entity identified by spaCy NER is encapsulated by the identified named entity

type. E.g., `<ORG>Indian Army</ORG>`.

• Dependency children of each verb are identified which are related to the verb with key dependency relations such as *nsubj* (nominal subject), *dobj* (direct object), and *nsubjpass* (passive nominal subject). The entire dependency subtree rooted at such children are encapsulated by the tags of the form `<DepRel_VERB>`. E.g., `<dobj_arrested>`··· `</dobj_arrested>`

• For each preposition, the tags of the form `<prep_PREP_PARENT>` are added to encapsulate the entire prepositional phrase modifying a noun or verb parent. E.g., `<prep_of_terrorists>`···`</prep_of_terrorists>`

• Complete dependency subtree rooted at each noun and verb is encapsulated using the tags of the form `<DNP_NOUN>` and `<DNP_VERB>`, respectively. E.g., `<DNP_terrorists>` and `<DVP_arrested>`

• If a noun or a verb is *negated* (i.e., having a child with dependency relation *neg* such as `no`, `not`, `never`), then the complete dependency subtree rooted at that noun or verb is encapsulated using the tags of the form `<NEG_NOUN>` and `<NEG_VERB>`.

• In addition, we add tags in the ETF text which indicate presence in gazetteers of multiple types.

2.2 Patterns Specification Language

We have designed a simple patterns specification language for writing linguistic rules for extracting entity mentions. Each pattern has the following important attributes:

• **Pattern ID**: An unique integer identifier.

• **Pattern properties**: Specify whether the pattern is case-sensitive, whether it is to be applied on the ETF text or plain text, and whether it is to be applied sentence-by-sentence or on the entire text.

• **MainRegex**: A valid regular expression pattern containing at least one *named group*[1]. Entity types to be extracted are used as named groups. This pattern is applied against the input text iteratively. If the pattern is matched successfully, then for each named group in the pattern, the text matched for that particular group is extracted as an entity mention of the corresponding entity type.

• **OuterRegex** (optional): If specified, OuterRegex is a valid regular expression with a named group with name *select*. MainRegex (defined above) gets matched only on the part of the input text selected by the *select* group of the OuterRegex. If OuterRegex not specified, then the MainRegex gets matched on the entire input text.

[1]`docs.python.org/3/howto/regex.html`

Table 1 shows an example of a linguistic pattern which extracts entity mentions of type Criminal from news articles. This pattern tries to extract names of the criminals by identifying person names within the direct object phrase of the verbs `arrested` or `detained` (see the illustration in Table 1).

Facilities in the patterns specification language for more powerful and effective patterns:

• **Embedded variables** can be used in a linguistic pattern (MainRegex or OuterRegex), using the syntax $<<< VAR >>>$ where VAR is defined separately as a regular expression. Such variables are expanded automatically within the pattern to get the final regular expression pattern.

• **Embedded entity types** can be used in a linguistic pattern (MainRegex or OuterRegex, using the syntax [[[ENTITY_TYPE]]]) where these are replaced with actual entity mentions of that entity type extracted in the same text by *earlier* patterns (i.e., patterns with lower PatternID).

3 Linguistic Rules for Weak Supervision

For extracting entity mentions, sequence labelling techniques such as Conditional Random Fields (CRF) and Long Short-term Memory (LSTM) networks are widely used. These supervised techniques need a significant number of annotated sentences for training to achieve desirable extraction accuracy. For any domain-specific entity type, creating a dataset of such annotated sentences involves significant time, manual efforts and cost.

We propose to augment a small training dataset (L) for such entity extraction task with unlabelled data (U) where labels are automatically obtained using linguistic patterns. A large number of additional sentences can be labelled in this way without any extra time and cost. Although, some manual efforts and expertise are needed for designing the linguistic patterns, the efforts are significantly less as compared to manually annotating a large corpus. The patterns are designed in such a way that each pattern is a high-precision pattern. We observed that a sequence labelling model trained using $L \cup U$ achieves better entity extraction performance as compared to a model trained using only L. Also the supervised sequence labelling model does not learn to imitate the linguistic rules exactly because:

Different feature views: The linguistic rules and a supervised sequence labelling model use two different feature views, similar to Co-training. Although

Text: `Indian Army arrested two terrorists of Al-Badr terror outfit namely Zahid Sheikh and Shareefudin Ahanger in connection with the murder of Danish Manzoor.`
ETF: `<nsubj_arrested><DVP_arrested><DNP_army><ORG><NNP>Indian </NNP><NNP>Army </NNP></ORG></DNP_army></nsubj_arrested> <VBD>arrested </VBD><dobj_arrested><DNP_terrorists><CARDINAL><CD>two </CD></CARDINAL><NNS>terrorists </NNS><prep_of_terrorists><IN>of </IN><pobj_of><DNP_outfit><CARDINAL><NNP>Al-Badr </NNP></CARDINAL><NN>terror </NN><NN>outfit </NN><RB>namely </RB><appos_outfit><DNP_sheikh><PERSON><NNP>Zahid </NNP><NNP>Sheikh </NNP></PERSON><CC>and </CC><DNP_ahanger><PERSON><NNP>Shareefudin </NNP><NNP>Ahanger </NNP></PERSON></DNP_ahanger></DNP_sheikh></DNP_outfit></DNP_terrorists></appos_outfit></pobj_of></prep_of_terrorists> </dobj_arrested><prep_in_arrested><IN>in </IN><pobj_in><DNP_connection><NN>connection </NN><prep_with_connection><IN>with </IN><pobj_with><DNP_murder><DT>the </DT><NN>murder </NN><prep_of_murder> <IN>of </IN><pobj_of><DNP_manzoor><NORP><NNP>Danish </NNP></NORP><PERSON><NNP>Manzoor </NNP></PERSON> </DNP_manzoor></DNP_murder></DNP_connection></pobj_of></prep_of_murder></pobj_with></prep_with_connection> </pobj_in></prep_in_arrested><.>. </.></DVP_arrested>`
Pattern ID: 3; **Pattern properties**: **N** (not case-sensitive), **E** (to be applied on ETF), **S** (to be applied sentence-by-sentence) **MainRegex**: `<PERSON>(?P<Criminal>.*?)</PERSON>` **OuterRegex**: `<dobj_(arrested|detained)>(?P<select>.*?)</dobj_(arrested|detained)>`
OuterRegex match for the *select* named group: `<DNP_terrorists><CARDINAL><CD>two </CD></CARDINAL><NNS>terrorists </NNS><prep_of_terrorists><IN>of </IN><pobj_of><DNP_outfit><CARDINAL><NNP>Al-Badr </NNP></CARDINAL><NN>terror </NN><NN>outfit </NN><RB>namely </RB><appos_outfit><DNP_sheikh><PERSON><NNP>Zahid </NNP><NNP>Sheikh </NNP></PERSON><CC>and </CC><DNP_ahanger><PERSON><NNP>Shareefudin </NNP><NNP>Ahanger </NNP></PERSON></DNP_ahanger></DNP_sheikh></DNP_outfit></DNP_terrorists></appos_outfit></pobj_of></prep_of_terrorists>`
MainRegex matches for the named group *Criminal*: First match: `<NNP>Zahid </NNP><NNP>Sheikh </NNP>`; Second match: `<NNP>Shareefudin </NNP><NNP>Ahanger </NNP>`

Table 1: Example of ETF text and a linguistic pattern matched against the ETF text

the feature views are not mutually exclusive, there are major differences. Most of the linguistic rules use dependency parsing information, which is not used by the sequence labelling model.

Multi-sentence patterns: Some of the linguistic patterns have multi-sentence scope, i.e., they use the context information which is outside the sentence from which an entity mention is identified. However, the sequence labeller processes only one sentence at a time, and hence it can not use any context information outside the current sentence. This enables the sequence labeller to learn additional features from the current sentence itself.

4 Related Work

The most relevant line of work for our linguistic patterns is Semgrex (Chambers et al., 2007), TRegex (Levy and Andrew, 2006), and spaCy Rule-based matching[2]. Semgrex and TRegex allow users to write patterns on dependency and constituency trees, respectively. The patterns are based on regular expression matching for nodes (tokens) and various relationships between the nodes. Rule-based matching provided by spaCy allows users to write regular expression patterns for token-level matching but for more complex rules, Python scripting is necessary. Our patterns specification language is specified purely in terms of regular expressions and allows users to write very powerful patterns using facilities such as MainRegex-OuterRegex combination (e.g., PatternID=3 in Table 2) and embedded entity types (e.g., PatternID=4 in Table 2).

A form of indirect supervision is *distant supervision* (Mintz et al., 2009) where a knowledge base is used to automatically create an annotated dataset. Recently, Snorkel framework (Ratner et al., 2017) was proposed to combine multiple weak supervision sources. However, it is not easily adaptable for sequence labelling tasks. Lison et al. (2020) proposed an entity extraction technique using weak supervision from multiple labelling functions such as entity extraction models trained on other domains, gazettes, heuristic functions etc. In our case, for a domain-specific entity like Risk entity (introduced in the next section), labelling functions based on other domains, gazettes or knowledge bases are not feasible. However, two recent approaches (Safranchik et al., 2020; Liang et al., 2020) look promising for our problem setting where linguistic rules are used for weak supervision for entity extraction and we plan to explore them as future work.

5 Application

Weak supervision using linguistic patterns is especially useful for extraction of domain-specific entity types for which obtaining or creating training data is costly. Hence, we demonstrate its effectiveness for extraction of mentions of one

[2] spacy.io/usage/rule-based-matching

Variable definitions (only partial patterns are shown due to space constraints):

```
NEGATIVE_NOUNS:=((un|non)\W*availability|breach(es)?|discrepanc(y|ies)|lack|delays?|slip(pages?)?|over\W*run···
RIGHT_BOUNDARY:=((,|\.|:)[ ]|\b(due|because|hence|which|who|that|based|if|may)[ ]|\bto[ ][^]*<VB[A-Z]?>)
NEGATIVE_VERBS:=(pending|(impact|delay|affect|hinder|hamper|disrupt)(ed|s|ing)?)
POSITIVE_VERBS:=(developed|maintained|installed|completed|followed|complied|tracked|updated|approved|defined···
```

Extraction patterns (above VARIABLEs are included using <<<VARIABLE>>>, the extracted entity mentions are shown in square brackets):

```
-PatternID=1 MainRegex: <DNP_<<<NEGATIVE_NOUNS>>>>(?P<Risk>((?!\bno[ ]).)*?[ ]((?!\bno[ ]).)*?[ ]((?!\bno[ ]).)*?)
(<<<RIGHT_BOUNDARY>>>|</DNP_<<<NEGATIVE_NOUNS>>>>)
    // E.g., Frequent changes in Tech Stack might lead to [delivery slippage].
-PatternID=2 MainRegex: <nsubj_<<<NEGATIVE_VERBS>>>>(?P<Risk>.*?)(<<<RIGHT_BOUNDARY>>>|</nsubj_<<<NEGATIVE_VERBS>>>>)
    // E.g., [Lack of right combination of skills in resources] may impact the timelines of the project delivery.
-PatternID=3 MainRegex: <nsubjpass_<<<POSITIVE_VERBS>>>>(?P<Risk>.*?\b(not|nt|no)[ ].*?\b<<<POSITIVE_VERBS>>>[ ])
OuterRegex: ^(?P<select>.*?)</NEG_<<<POSITIVE_VERBS>>>>
    // E.g., It was observed that [assessment on data privacy was not completed].
-PatternID=4 MainRegex: <nsubj_leads?>(?P<Risk>.*?)</nsubj_leads?>.*?\bleads?[ ][^ ]*?\bto[ ][^ ]*?[[[Risk]]]
    // E.g., [Frequent changes in Tech Stack] might lead to delivery slippage.
-PatternID=5 (Multi-sentence)
MainRegex: <DNP_[A-Za-z]+>(?P<Risk>((?!<SENT>|</SENT>|<<<RIGHT_BOUNDARY>>>).)*)</DNP_[A-Za-z]+>
OuterRegex: \brisks?[ ][^ ]*?:[ ][^ ]*?</SENT>(?P<select>.*?)$
```

Table 2: Representative linguistic patterns for extraction of Risk entity mentions.

such type – Risk. In large IT services organizations, thousands of projects are going on simultaneously. These projects are routinely audited and as a part of this process, auditors also write their opinion about each project as an audit summary. One of the most important piece of information in these audit summaries is potential *Risks* that the project is facing or may face in near future. We define Risk as an entity type which is any undesirable factor which may have an adverse effect on project objectives or outcomes. E.g., `impacting timelines of the project delivery`, `unavailability of skilled resources`. It is important to note that Risk mentions can be not only noun phrases but also verb phrases. Hence, extraction of Risk mentions is challenging compared to the traditional Named Entity Recognition (NER) task where the entity types (such as PERSON or ORG) are mentioned in the form of noun phrases only.

Table 2 shows some linguistic patterns designed for extraction of Risk entity mentions along with examples of sentences and extracted mentions. **PatternID=4** uses the embedded entity type [[[Risk]]] where the final pattern dynamically substitutes Risk extractions by earlier patterns (`delivery slippage` in this case which is already extracted by the first pattern). **PatternID=5** is a multi-sentence pattern which first identifies a list of sentences immediately followed by "`risks:`" (using OuterRegex) and then extracts noun phrases from such sentences as Risk mentions (using MainRegex).

We used a dataset of 3804 audit summaries consisting of 8046 sentences. We manually annotated Risk entity mentions in 700 of these and used 500 as our training set (L) and remaining 200 as our evaluation set. 3104 unlabelled audit summaries (U) were used to augment the manually annotated training set L using the entity mentions identified in U by the linguistic rules. Table 3 shows the overall entity extraction performance on the evaluation set, using CRF and BiLSTM-CRF (Huang et al., 2015) models. It can be observed that the models trained on $L \cup U$ clearly outperform the models trained only on L as well as only rules-based extraction. Consider the sentence: `The SIT and UAT environment is same, this may impact the quality of the deliverables`. Here, only the BiLSTM-CRF model trained using $L \cup U$ was able to extract the Risk mention `The SIT and UAT environment is same` which was neither extracted by the linguistic rules nor by the model trained only on L.

Technique	**P**	**R**	**F1**
Only Linguistic Rules	0.73	0.38	0.50
CRF trained using L	0.60	0.28	0.38
BiLSTM-CRF trained using L	0.41	0.38	0.39
CRF trained using $L \cup U$	0.59	0.37	0.46
BiLSTM-CRF trained using $L \cup U$	0.65	0.54	**0.59**

Table 3: Extraction accuracy for RISK mentions using the evaluation dataset of 200 Audit summaries

6 Conclusions

We proposed a powerful patterns specification language for specifying linguistic rules for entity extraction which are matched against ETF text. The language is also generalizable to encode linguistic knowledge for relation and event extraction. We demonstrated how an entity extraction system can be quickly built for a domain-specific entity type Risk where a small manually annotated dataset is augmented with a large automatically labelled dataset using linguistic knowledge.

References

Nathanael Chambers, Daniel Cer, Trond Grenager, David Hall, Chloe Kiddon, Bill MacCartney, Marie-Catherine De Marneffe, Daniel Ramage, Eric Yeh, and Christopher D Manning. 2007. Learning alignments and leveraging natural logic. In *Proceedings of the ACL-PASCAL Workshop on Textual Entailment and Paraphrasing*, pages 165–170.

Matthew Honnibal and Ines Montani. 2017. spacy 2: Natural language understanding with bloom embeddings, convolutional neural networks and incremental parsing. https://spacy.io/.

Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional lstm-crf models for sequence tagging. *arXiv preprint arXiv:1508.01991*.

Roger Levy and Galen Andrew. 2006. Tregex and tsurgeon: tools for querying and manipulating tree data structures. In *LREC*, pages 2231–2234. Citeseer.

Jing Li, Aixin Sun, Jianglei Han, and Chenliang Li. 2020. A survey on deep learning for named entity recognition. *IEEE TRANSACTIONS ON KNOWLEDGE AND DATA ENGINEERING*, page 1.

Chen Liang, Yue Yu, Haoming Jiang, Siawpeng Er, Ruijia Wang, Tuo Zhao, and Chao Zhang. 2020. Bond: Bert-assisted open-domain named entity recognition with distant supervision. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 1054–1064.

Pierre Lison, Aliaksandr Hubin, Jeremy Barnes, and Samia Touileb. 2020. Named entity recognition without labelled data: A weak supervision approach. *arXiv preprint arXiv:2004.14723*.

Mike Mintz, Steven Bills, Rion Snow, and Dan Jurafsky. 2009. Distant supervision for relation extraction without labeled data. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 1003–1011.

G. K. Palshikar. 2012. Techniques for named entity recognition: A survey. In *Collaboration and the Semantic Web: Social Networks, Knowledge Networks and Knowledge Resources*, pages 191–217. IGI Global.

S. Pawar, G.K. Palshikar, and P. Bhattacharyya. 2017. Relation extraction: A survey. In *arXiv:1712.05191*.

Alexander Ratner, Stephen H Bach, Henry Ehrenberg, Jason Fries, Sen Wu, and Christopher Ré. 2017. Snorkel: Rapid training data creation with weak supervision. In *Proceedings of the VLDB Endowment. International Conference on Very Large Data Bases*, volume 11, page 269. NIH Public Access.

Esteban Safranchik, Shiying Luo, Stephen H Bach, Elaheh Raisi, Stephen H Bach, Stephen H Bach, Daniel Rodriguez, Yintao Liu, Chong Luo, Haidong Shao, et al. 2020. Weakly supervised sequence tagging from noisy rules. In *AAAI*, pages 5570–5578.