Automated Acquisition of Patterns for Coding Political Event Data: Two Case Studies

Peter Makarov
Institute of Computational Linguistics
University of Zurich, Switzerland
makarov@cl.uzh.ch

Abstract

We present a simple approach to the generation and labeling of extraction patterns for coding political event data, an important task in computational social science. We use weak supervision to identify pattern candidates and learn distributed representations for them. Given seed extraction patterns from existing pattern dictionaries, we use label propagation to label pattern candidates. We present two case studies. i) We derive patterns of acceptable quality for a number of international relations & conflicts categories using pattern candidates of O'Connor et al. (2013). ii) We derive patterns for coding protest events that outperform an established set of TABARI / PETRARCH hand-crafted patterns.

1 Introduction

Social scientists work with datasets of interactions between political actors (*political events*), which they extract manually or automatically (*code*) from large quantities of news text (Figure 1). The automated coding of political events, which dates back to the early 1990s (Gerner et al., 1994), is commonly performed using pattern matching with large manually compiled dictionaries of actor names and event patterns. Syntactic parsing is widely used to guide the application of patterns.

Example	U.S. military chief General Colin Powell said on Wednesday NATO would need to remain strong.
Event type	MAKE STATEMENT, GENERIC (010)
Example	Kenyan President Daniel Arap Moi on Monday urged Uganda to repatriate "all Kenyan criminals hiding there" to face trial, accusing them of killing Kenyan policemen in cross-border raids recently.
Event type	APPEAL FOR MATERIAL COOPERATION (021)

Example	Austrian unions blocked three motorways into the capital Vienna on Monday to protest government plans to reform the country's pension system.
Event type	OBSTRUCT PASSAGE, BLOCK (144)
Example	A small eastern German company on Wednesday became the first to announce a boycott by an American company over Berlin's refusal to back the U.S. administration's moves to disarm Iraq militarily.
Event type	CONDUCT STRIKE OR BOYCOTT (143)

Figure 1: Examples of political events and their CAMEO event categories (§ 2): IR events above and protest events below. The IR examples are from the CAMEO codebook (Schrodt, 2012). In the examples, the source actor is in red, the event in green, and the target actor in blue.

Despite their simplicity, pattern-based coding systems are as good as trained human coders at predicting event types (King and Lowe, 2003) and have been found sufficiently accurate for near real-time event monitoring (O'Brien, 2010). Compared to statistical systems for event extraction common in NLP (Ahn, 2006), one advantage of pattern-based event coding is that coding decisions are transparent and readily examinable, being triggered by the matching of specific patterns. Yet, manual pattern construction is

Proceedings of Workshop on Computational Linguistics for Cultural Heritage, Social Sciences, Humanities and Literature, pages 103–112
Santa Fe, New Mexico, USA, August 25, 2018.

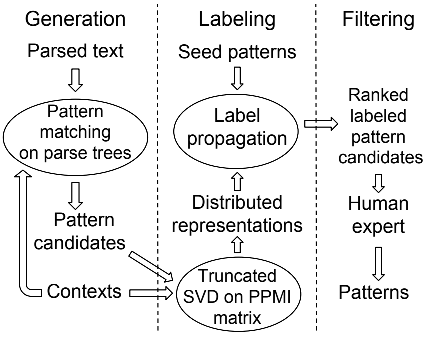

Generation
Labeling
Filtering
Parsed text
Pattern matching on parse trees
Pattern candidates
Contexts
Seed patterns
Label propagation
Distributed representations
Truncated SVD on PPMI matrix
Ranked labeled pattern candidates
Human expert
Patterns

Figure 2: Overview of proposed pipeline: automated pattern acquisition and subsequent filtering of pattern candidates by a human expert.

$reject \xrightarrow{\text{dobj}} criticism \xrightarrow{\text{prep\_of}}$	$blacklist \xrightarrow{\text{dobj}}$
$postpone \xrightarrow{\text{dobj}} visit \xrightarrow{\text{poss}}$	$kill \xrightarrow{\text{prep\_as}}$
$pelt \xrightarrow{\text{dobj}} stones$	$pelt \xrightarrow{\text{prep\_at}} police$
$attack \xrightarrow{\text{nsubjpass}} office$	$remain \xrightarrow{\text{prep\_despite}} change$
$receive \xrightarrow{\text{dobj}} dobj$	$protect \xrightarrow{\text{prep\_from}} violence$

Figure 3: Examples of pattern candidates: IR domain above (O'Connor et al., 2013), protest below.

prohibitively costly (Schrodt, 2006). Patterns are not easily portable across domains, and any adaptation to a new domain requires extensive human effort.

In this paper, we show how the automated acquisition, that is generation and labeling, of extraction patterns can be applied to the problem of the machine coding of political events. The goal is, on the one hand, to reduce human effort associated with pattern construction and, on the other, to show how to increase recall, which is often low in rule-based systems. By automatically generating pattern candidates and labeling them in a semi-supervised way, we produce many noisy patterns. Before any of them gets added to a pattern-based coder, they should be inspected by a human expert.

Our contribution is as follows: We combine ideas from traditional pattern acquisition by bootstrapping (Huang and Riloff, 2013), the more recent approaches to the grouping of semantically related patterns based on distributed representations (Krause et al., 2015; Batista et al., 2015), and semi-supervised labeling for lexicon induction (Hamilton et al., 2016; Rao and Ravichandran, 2009; Takamura et al., 2007) into a simple recipe for pattern construction, of practical interest to social scientists (Figure 2). We demonstrate the effectiveness of the approach by applying it to the important domains of international relations and conflicts (IR) and of protest events.

2 Related Work

There exists a whole family of political event coding systems that rely on dictionaries of patterns: TABARI, PETRARCH, PETRARCH2[1] (Norris et al., 2017), and VRA-Reader[2] (King and Lowe, 2003). One popular ontology of political events, primarily in the IR domain, is CAMEO (Gerner et al., 2002). It defines twenty broad categories, e.g. Reduce Diplomatic Relations or Investigate. Each topmost category is further divided into more specific subtypes, e.g. Investigate War Crimes. TABARI, PETRARCH, and PETRARCH2 come with extensive dictionaries of patterns that map verb phrases to CAMEO categories. Recently, there has been lot of interest in applying statistical learning to the coding of political events (Beieler, 2016; Hanna, 2017; Nardulli et al., 2015). O'Connor et al. (2013) present an unsupervised Bayesian coder, which models the gradual change in the types of events between actors over time.

Automated pattern acquisition has been a central topic in information extraction by pattern matching (Yangarber et al., 2000; Riloff and Jones, 1999) and is primarily associated with bootstrapping, a set of heuristic methods that establish similarity between patterns based on their occurrence in a small number of contexts. Huang and Riloff (2013) bootstrap verb-phrase patterns for protest event extraction by exploiting their co-occurrence with collective terms like "workers", "activists". Krause et al. (2015) explore the occurrence of event patterns in all contexts and train a feedforward neural network to produce event pattern embeddings.

The semi-supervised method of label propagation in a lexical similarity graph whose edge weights are computed from rich distributed representations has been used extensively for the derivation of large-scale polarity lexica (Hamilton et al., 2016; Velikovich et al., 2010). Various efficient techniques have become

[1] `https://github.com/openeventdata/petrarch2`

[2] `http://vranet.com`

available for obtaining powerful distributed representations of linguistic entities (Mikolov et al., 2013; Pennington et al., 2014; Cotterell et al., 2017). Levy et al. (2015) repopularize truncated singular value decomposition (SVD) in the context of word embeddings.

3 Method

This and the next section present the method. First, we generate pattern candidates from dependency parses (§ 3.1), then compute their distributed representations and label using label propagation (§ 3.2).

3.1 Generation of pattern candidates

A *pattern* is a most simple classifier (r, t) that consists of a regular expression r and a type t. If r matches some substring v of input text x, then x gets classified with t and v is the textual evidence of t in x. A *pattern candidate* is a regular expression. Given a set of types $\{t_1, \dots, t_m\}$, an automated pattern acquisition method identifies sets of pattern candidates $\{P_1, \dots, P_m\}$ such that P_i are likely correct classifiers for type t_i. To apply P_i to coding, one can either simply use all P_i as patterns for t_i (i.e. $\{(p, t_i) : p \in P_i\}$) or have a human expert examine P_i and build patterns from reliable pattern candidates only.

We follow the standard practice of using dependency paths as pattern candidates (Stevenson and Greenwood, 2006). A *dependency path* is a path through a syntactic dependency structure, i.e. an alternating sequence of labeled edges and nodes $(n_0, e_0, n_1, e_1, \dots)$ such that each e_i connects n_{i-1} and n_i and e_i can be either left-to-right or right-to-left. A path can end with either a node or an edge. For example, $protester \xleftarrow{\text{subj}} pelt \xrightarrow{\text{dobj}} stone$ is a dependency path. We shall assume that dependency paths use lemmas instead of tokens. Whenever this is not the case we shall explicitly state it.

Being in a specific dependency relation with likely actor expressions is what makes an arbitrary dependency path a pattern candidate for some type. The exact nature of pattern candidates is dictated by weak supervision, which is specific to a domain.

We shall now detail the generation of pattern candidates for the protest and IR domains. IR pattern candidates are due to O'Connor et al. (2013) and are generated from paths connecting source and target actor expressions. Protest pattern candidates are predicates of collective actor expressions, an idea for pattern bootstrapping due to Huang and Riloff (2013).

Protest events. We work with 1.8M newswire documents[3] downloaded from the LexisNexis data service[4] using a search query with common protest-related keywords.[5] We process the documents with the Stanford CoreNLP toolkit (Manning et al., 2014). We build pattern candidate generation on the following observation (Huang and Riloff, 2013): Protest events are typically collective actions, and therefore, when they are expressed as verbs, their semantic agent will likely be a plural noun, e.g. "Workers took to the streets", "A group of students clashed with the police". This suggests a simple procedure: We find all plural nouns and traverse dependency trees collecting all predicates of which the plural nouns are agents.[6]

We identify all plural common nouns (i.e. NNS-tagged tokens)[7] and then traverse collapsed and

[3] Agence France Press, Deutsche Presse-Agentur (German Press Agency), BBC World from 2000-2014.

[4] `https://www.lexisnexis.com`

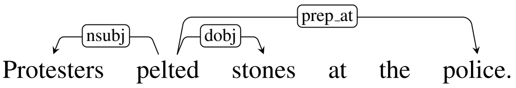
[5] initiative OR referendum OR petition! OR signature! OR campaign! OR protest! OR demonstrat! OR manifest! OR marche! OR marchi! OR parade OR rall! OR picket! OR (human chain) OR riot! OR affray OR festival OR ceremony OR (street theatre) OR (road show) OR vigil OR strike! OR boycott! OR block! OR sit-in OR squat! OR mutin! OR bomb! OR firebomb! OR molotov OR graffiti OR assault OR attack OR arson OR incendiar! OR (fire I/1 raising) OR (set AND ablaze) OR landmine OR sabot! OR hostage! OR assassinat! OR shot OR murdered OR killed (Kriesi et al., 2012)

[6] Our procedure is different in many important ways from that of Huang and Riloff (2013). We identify a larger set of syntactic constructions that contribute pattern candidates, including passives, relative clauses, and more types of verbal adjuncts. Unlike Huang and Riloff (2013), we do not distinguish between event phrases and purpose phrases (e.g. in the sentence "Workers took to the streets to demand better working conditions.", "took to the streets" would be an event phrase and "to demand better working conditions" a purpose phrase). We consider purpose phrases a subset of event phrases, partly because often the same predicate occurs as both. Further, the original proposal tries to identify purpose phrases, like in the example sentence above, with the help of the *xcomp* relation, which is in fact a parser error: Purpose phrases are not arguments but *advcl* dependents.

[7] In fact, we could have also included plural proper nouns (NNPS-tagged tokens).

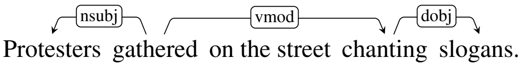
CCprocessed dependencies (De Marneffe and Manning, 2008). A pattern candidate is a path from a verb to its (direct or prepositional) object or prepositional adjunct (i.e. a dependent in any relation matching *dobj|prep_**). However, if the object or adjunct is an named entity (NE), we store the NE tag and discard the lemma. If a sentence matches one or more of the following cases, we extract pattern candidates and update the statistics of co-occurrence of plural common nouns and pattern candidates:

1. A plural common noun is a *nsubj* dependent (i.e. a subject of an active verb). E.g. in the sentence

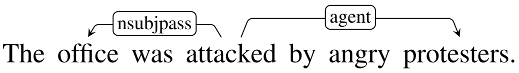


   we identify candidates $pelt \xrightarrow{\text{dobj}} stone$ and $pelt \xrightarrow{\text{prep\_at}} police$.

2. The main verb has a plural common noun subject and a *xcomp* or *vmod* dependent (i.e. a non-finite verbal complement or adjunct).[8] The non-finite dependent produces pattern candidates, e.g. $chant \xrightarrow{\text{dobj}} slogan$ from



3. A plural common noun is an *agent* dependent. The verb that governs it produces a pattern candidate together with its *nsubjpass* dependent, or *rcmod* or *vmod* head. The former accounts for passives, the latter for finite and non-finite relative clauses, e.g.



After excluding infrequent pattern candidates ($count < 15$) and nouns ($count < 5$), we obtain 72K unique pattern candidates and 11K plural common nouns. Together, they produce 3.6M pattern-noun samples (Figure 3).

International relations. Here, we re-use pattern candidates derived by O'Connor et al. (2013) from the English Gigaword corpus of newswire documents (Parker et al., 2009). A pattern candidate is a dependency path connecting source and target actor expressions. Actor expressions are identified with the help of the TABARI actor dictionaries. An actor expression is the minimal noun phrase containing a TABARI actor expression: a proper noun or adjective. Pattern candidates contain at most four notional words. The source actor must be a *nsubj* or *agent* dependent. Some dependency relations are not allowed in the patterns, e.g. *det* or *conj*. For further details, we refer the reader to the paper. Each pattern candidate is associated with a source-target actor pair (*dyad*) and the publication date of the news report. After filtering low-frequency pattern candidates and dyads, O'Connor et al. obtain 366K data samples featuring 421 dyads and 10K unique pattern candidates.

3.2 Labeling of pattern candidates

The semi-supervised labeling of pattern candidates requires a semantic similarity metric on the set of pattern candidates. Cosine similarity between vectors in some vector space, which somehow capture the semantics of the corresponding linguistic entities (e.g. words or patterns), is one very common similarity metric. The derivation of vector representations leverages the counts of how many times the linguistic entities occur in some specific *contexts* in the data. For example, in the derivation of word vectors, the contexts are simply neighboring words.

[8] We do not include *advcl* for simplicity: *advcl* dependents can have their own subjects different from the subject of the main verb.

Step 1: PPMI matrix Define matrix $\mathbf{M} \in \mathbb{R}^{m \times k}$ of PPMI scores, where m is the number of pattern candidates and k is the number of contexts, as

$$m_{qc} = \max\Big(\log \frac{\hat{p}(q,c)}{\hat{p}(q)\, \hat{p}_\alpha(c)},\ 0 \Big),$$

for pattern candidate q and context c. $\hat{p}(q,c)$ and $\hat{p}(q)$ are empirical distributions, $\hat{p}_\alpha(c)$ is a smoothed context distribution defined as $\hat{p}_\alpha(c) = \frac{\#(c)^\alpha}{\sum_c \#(c)^\alpha}$, where $\alpha \in \mathbb{R}$ and $\#(c)$ is the count of c in the data.

Step 2: Dimensionality reduction Perform singular value decomposition

$$\mathbf{M} = \mathbf{U} \cdot \mathbf{\Sigma} \cdot \mathbf{V}^\top$$

Rows vectors of matrix $\mathbf{U}^{\text{tr}}$ truncated at the first l columns are distributed representations of pattern candidates.

Procedure 1: Derivation of distributed representations for pattern candidates.

Contexts. For event patterns, contexts can be actor expressions or other patterns that occur with the same actor expressions. For protest patterns, we use plural common nouns as contexts.

For IR patterns, similar to O'Connor et al. (2013) and Krause et al. (2015), we assume that a dyad and a time span induce a context: Patterns that occur with the same dyad in the texts from the same time span become contexts for one another. In our experiments, we take the time span to be a single day—the publication date. Thus, given a (dyad, publication date) tuple t, pattern candidates p and q that occur with t, $p \neq q$, are counted once as p being a pattern and q its context and once as q being a pattern and p its context. This is identical to Krause et al. (2015).

Distributed representations. We choose to derive distributed representations for pattern candidates by first constructing a pattern-context matrix $\mathbf{M}$ of positive pointwise mutual information (PPMI) scores (Step 1 of Procedure 1) and then performing SVD on $\mathbf{M}$ (Step 2 of Procedure 1). To this end, we use the `Hyperwords`[9] package. We apply context distribution smoothing with $\alpha = 0.75$ and retain only the pattern-to-latent-factors matrix $\mathbf{U}^{\text{tr}}$ (Levy et al., 2015) truncated at 500 columns.

At this point, we can already cluster pattern candidates by e.g. exploring pattern frequencies (Krause et al., 2015). Fortunately, we have at our disposal the TABARI dictionary of labeled event patterns, which we can use in a semi-supervised learning procedure to label new patterns. This follows closely the approach of Hamilton et al. (2016), and we re-use much of their code.[10]

Similarity graph. First, we construct a weighted undirected graph of pattern similarity. Pattern candidates are nodes, and the weights of the edges $\mathbf{W}$ are computed using angular similarity, which turns cosine similarity into a distance metric. Self-loops are disallowed (Step 1 of Procedure 2). For efficiency, for each pattern candidate, we keep connections to only twenty five most similar nodes; all other edge weights are set to zero.

Label propagation. We apply the semi-supervised label propagation algorithm of Zhou et al. (2004) (Step 2 of Procedure 2) and use the TABARI verb pattern dictionary to identify seed patterns. For experiments with IR patterns, we find seeds for all but one of the topmost CAMEO categories (Mass Violence), with the number of seeds per category ranging from one to seventy-four. We identify thirty four seeds among protest event pattern candidates.

4 Experiments

We next evaluate the quality of the labeled pattern candidates. Event patterns are intended as high-precision classifiers: The words making a pattern are chosen carefully to generate as few false positives as possible. Pattern-based classifiers are typically weak in recall as it may be difficult to construct sufficiently many unambiguous patterns. Thus, any good new pattern potentially contributes to higher recall. Does our automated approach produce good new patterns? An evaluation that we conduct for IR patterns

[9] `https://bitbucket.org/omerlevy/hyperwords`

[10] `https://github.com/williamleif/socialsent`

Step 1: Edge weight matrix Define edge weight matrix $\mathbf{W} \in \mathbb{R}^{m \times m}$ as

$$w_{pq} = \begin{cases} \arccos\big(-\cos\theta\big) & \text{if } p \neq q, \\ 0 & \text{otherwise} \end{cases}$$

where $\cos\theta$ is the cosine similarity between row vectors $\mathbf{u}^{\text{tr}}_{p,*}$ and $\mathbf{u}^{\text{tr}}_{q,*}$ of $\mathbf{U}^{\text{tr}}$, the distributed representations of patterns p and q.

Step 2: Label propagation (After Zhou et al. (2004)) Given some seed patterns, define matrix $\mathbf{F}^{(0)} \in \mathbb{R}^{m \times c}$, where m is the number of patterns and c is the number of categories, as

$$f^{(0)}_{pk} = \begin{cases} 1 & \text{if } p \text{ is a seed pattern for category } k, \\ 0 & \text{otherwise} \end{cases} \tag{1}$$

Define transition matrix $\mathbf{T} \in \mathbb{R}^{m \times m}$ as

$$t_{pq} = \frac{w_{pq}}{\sum_{r=0}^{m} w_{rq}} \tag{2}$$

Set $\beta \in [0, 1]$. Until convergence, iterate

$$\mathbf{F}^{(t+1)} = \beta \mathbf{T}\,\mathbf{F}^{(t)} + (1-\beta)\mathbf{F}^{(0)} \tag{3}$$

Let $\mathbf{F}^{(\infty)}$ be the label matrix after convergence is reached. Label unlabeled patterns q with $\hat{k} = \arg\max_k f^{(\infty)}_{qk}$.

Procedure 2: Labeling of pattern candidates by label propagation through pattern similarity graph.

aims at estimating the proportion of correct new patterns at various ranks, when ordered by label score $f^{(\infty)}_{qk}$. For protest events, we measure precision and recall on an annotated corpus.

International relations. We use IR patterns to code newswire documents from the LexisNexis data service. Out of twenty CAMEO categories, we randomly sample eight:

(A) four categories with the number of seed patterns greater than thirty: Engage in Diplomatic Cooperation, Fight/Assault,[11] Consult, Disapprove,

(B) two categories with ten to thirty seed patterns: Coerce, Reduce Relations, and

(C) two categories with fewer than ten seeds: Investigate, Exhibit Military Posture.

For each category k, we order the pattern candidates for k by label score $f^{(\infty)}_{qk}$ in descending order and randomly sample fifteen pattern candidates from among the first 50, 50-100, and 100-150 pattern candidates. We pair the sampled candidates with category labels and turn the resulting patterns to the TABARI / PETRARCH dictionary format.[12] We use PETRARCH to code actors and events with the help of these patterns. We also check that each event match respects the dependency path of the pattern. For each pattern, we randomly sample up to two sentences that it matches. This gives us a total of 551 sentences. To this, we add 130 sentences matched by thirteen patterns randomly sampled from the seeds of each category, with up to two sentences per pattern.

To estimate the proportion of correct new patterns, the author and one political science doctoral student check the predicted categories of all the sentences. The human coders try to indicate, whenever possible, whether the predicted category is incorrect for a reason other than the pattern assigning a wrong code. We exclude such cases from calculations. With this strategy, we aim to evaluate the (average stratified) precision of the patterns (i.e. their intended property as high-precision classifiers) irrespective of their frequency.[13] This is in contrast with estimating the precision of the entire pattern-based classifier, which would inevitably be dominated by high-frequency patterns.

[11] Here, we follow Boschee et al. (2013) and Boschee et al. (2015) in considering Fight and Assult a single topmost category.

[12] Somewhat surprisingly, above five percent of the patterns cannot be faithfully converted to PETRARCH format as they are headed by a noun and not a verb.

[13] An alternative strategy would be to manually inspect a list of patterns, which has a downside that we may overlook ambiguities that become apparent when we see a pattern in a sentence that it matches.

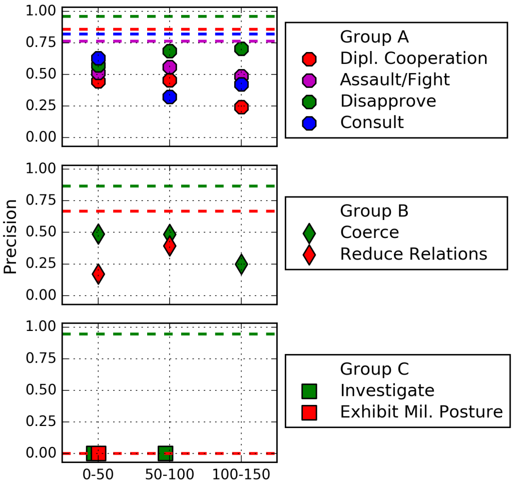


Figure 4: Stratified average pattern precision across CAMEO categories. Scores for patterns sampled from among the first 50 patterns, 50-100 patterns, 100-150 patterns. Dashed lines of same color mark average precision of seed patterns.

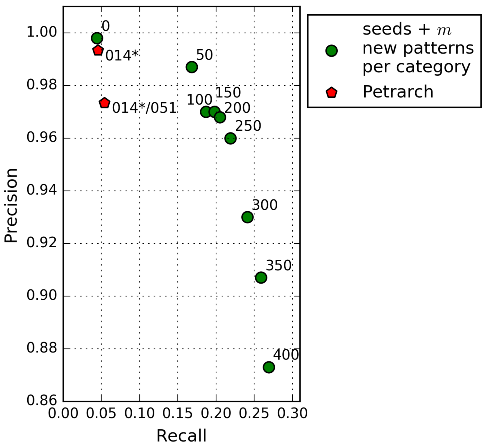


Figure 5: Precision and recall of seeds + m new patterns per protest subcategory and PETRARCH with default verb patterns and without actor coding. PETRARCH: 014* counting only protest events, 014*/051 counting protest events and events of subcategory (051) Rally Support For.

Figure 4 shows the results. The patterns for the categories with many seeds (groups **A** and **B**) perform well and compare favorably to seed patterns. Group **C** produce very few coded sentences, and no sentence is correctly coded. About eleven percent of all the sentences have been excluded. The most common causes for that are sentences from sports news (34%), hypothetical constructions (23%), negation (22%), and wrongly coded actors (12%). These sources of error are clearly failures not of the automatically generated patterns but the automated coder, e.g. its inability to take into account negation or check whether a pattern match is embedded under a modal verb. The inter-coder agreement for this evaluation is modest, with a Cohen's kappa of 0.64.

Protest events. We use the corpus of Makarov et al. (2016) of English Gigaword documents annotated with protest events and resembling standard benchmark datasets for event extraction, e.g. the ACE 2005 corpus (ACE, 2005). Unlike the IR patterns for which we resort to a complex evaluation strategy, with this corpus, we can directly and fully automatically estimate both precision and recall. We select sentences for which at least two of the coders code the same event. We only use sentences which feature event types that correspond to the five CAMEO protest codes.[14] We obtain a total of 572 labeled sentences. We randomly sample another 600 sentences of newswire text not from the corpus and use them to approximate the negative class, i.e. sentences without protest events.

Collective actions often do not mention a target actor directly (as in e.g. "protest against the anti-gay law"). Since an automated coder codes an event only if both source and target actors are matched, we choose to evaluate protest patterns without actor coding (otherwise, only very few events would be coded).

We note that some positive-class sentences cannot be coded without the knowledge of the context of the document. Another complication is that protest events are often referred to with standalone nouns like "demonstration" or "rally" and not verb phrases denoting protest actions, e.g. "the Florence demonstration was expected to be the biggest in the country" or "a group of rowdy youths broke away from the peaceful demonstration". Such cases, therefore, cannot be coded by a pattern-based coder that associates events with verb phrases. This suggests that the upper bound on recall in this evaluation is much below 100%.

[14] Subcategories of Protest (14*): Uncategorized Protest (140), Demonstration (141), Hunger Strike (142), Strike/Boycott (143), Obstruction of Passage (144), and Riots (145)

We compare the following conditions:

i) We code with seed patterns plus, for each of the five protest subcategories, the first m new patterns, when ranked by label score in descending order. We let m range from 0 to 400. We apply patterns by matching their dependency paths.

ii) As our baseline, we run PETRARCH with its default pattern dictionary (of a few hundreds of protest event patterns) and the actor coding function switched off. Additionally, we test a condition in which we consider category (051) Rally Support For as another protest category: Many protest events with a positive stance on an issue (as in "rallied for immigrant rights") end up being coded this code.

We manually check the events that either system finds in the negative-class sentences. We find eight sentences with protest events, which we then count as instances of the positive class.

The results indicate (Figure 5) that new patterns dramatically increase recall and precision remains high. Most matches by PETRARCH come from matching single verbs: "demonstrate", "protest", "rally". The new patterns, on the other hand, are more lexically diverse: Patterns that fire feature forty seven (at $m = 50$) to fifty five (at $m = 400$) different verbs.

5 Discussion

How do we justify the choices of the techniques in the pipeline? One finds various strategies in the literature for the generation of event and relation extraction patterns that typically employ as contexts nouns, especially proper nouns (Riloff and Jones, 1999; Carlson et al., 2010), although other kind of expressions also appear, e.g. verb phrases (Huang and Riloff, 2013). Although one can build patterns over words (Du and Yangarber, 2015), patterns over parse trees have been found useful abstractions (Sudo et al., 2003; Bunescu and Mooney, 2005; Stevenson and Greenwood, 2005). Likewise, other strategies for factorizing a pattern-context co-occurrence matrix can be employed. However, truncated SVD of a PPMI matrix provides competitive representations (Levy et al., 2015; Hamilton et al., 2016) and does not build in any assumptions about the nature of entities and contexts.

Pattern-based coding suffers from the simple logic of pattern application and errors in linguistic analyses. A more flexible and better performing approach is based on statistical learning (Boschee et al., 2013; Boschee et al., 2015). Syntactic patterns can be used as features in such a statistical event coding system. Syntactic information is important to systems solving a related task of semantic role labeling (Marcheggiani and Titov, 2017; Roth and Lapata, 2016; FitzGerald et al., 2015).

6 Conclusion

We present an approach for learning dictionaries of verb patterns for the coding of political events. The method uses pattern matching over dependency parse trees to identify pattern candidates, then computes distributed representations for them that define weights in a similarity graph. The labels of unlabeled pattern candidates are learned with a semi-supervised algorithm of label propagation through the resulting graph. New patterns evaluate favorably on two important domains of political interactions.

Acknowledgements

We would like to thank Simon Clematide and the anonymous reviewers. This work has been supported by European Research Council Grant No. 338875.

References

2005. The Automated Content Extraction 2005 (ACE 05) challenge. https://www.ldc.upenn.edu/collaborations/past-projects/ace/annotation-tasks-and-specifications.

David Ahn. 2006. The stages of event extraction. In *Proceedings of the Workshop on Annotating and Reasoning about Time and Events*, pages 1–8. ACL.

David S Batista, Bruno Martins, and Mário J Silva. 2015. Semi-supervised bootstrapping of relationship extractors with distributional semantics. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, Association for Computational Linguistics, Lisbon, Portugal*, pages 499–504.

John Beieler. 2016. Generating politically-relevant event data. *NLP+CSS 2016*.

Elizabeth Boschee, Premkumar Natarajan, and Ralph Weischedel. 2013. Automatic extraction of events from open source text for predictive forecasting. In *Handbook of Computational Approaches to Counterterrorism*, pages 51–67. Springer.

Elizabeth Boschee, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. BBN ACCENT Event Coding Evaluation.updated v01.pdf (ICEWS Coded Event Data). `http://dx.doi.org/10.7910/DVN/28075`.

Razvan C Bunescu and Raymond J Mooney. 2005. A shortest path dependency kernel for relation extraction. *HLT/EMNLP*, pages 724–731.

Andrew Carlson, Justin Betteridge, Richard C Wang, Estevam R Hruschka Jr, and Tom M Mitchell. 2010. Coupled semi-supervised learning for information extraction. In *Proceedings of the third ACM international conference on Web search and data mining*, pages 101–110. ACM.

Ryan Cotterell, Adam Poliak, Benjamin Van Durme, and Jason Eisner. 2017. Explaining and generalizing skip-gram through exponential family principal component analysis. *EACL 2017*, 175.

Marie-Catherine De Marneffe and Christopher D Manning. 2008. Stanford typed dependencies manual. Technical report, Technical report, Stanford University.

Mian Du and Roman Yangarber. 2015. Acquisition of domain-specific patterns for single document summarization and information extraction. In *The Second International Conference on Artificial Intelligence and Pattern Recognition (AIPR2015)*, page 30.

Nicholas FitzGerald, Oscar Täckström, Kuzman Ganchev, and Dipanjan Das. 2015. Semantic role labeling with neural network factors. In *EMNLP*, pages 960–970.

Deborah J Gerner, Philip A Schrodt, Ronald A Francisco, and Judith L Weddle. 1994. Machine coding of event data using regional and international sources. *International Studies Quarterly*, pages 91–119.

Deborah J Gerner, Philip A Schrodt, Omür Yilmaz, and Rajaa Abu-Jabr. 2002. Conflict and mediation event observations (CAMEO): A new event data framework for the analysis of foreign policy interactions. *International Studies Association, New Orleans*.

William L Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. 2016. Inducing domain-specific sentiment lexicons from unlabeled corpora. *arXiv preprint arXiv:1606.02820*.

Alex Hanna. 2017. MPEDS: Automating the Generation of Protest Event Data. Available at SSRN: `https://osf.io/preprints/socarxiv/xuqmv`.

Ruihong Huang and Ellen Riloff. 2013. Multi-faceted event recognition with bootstrapped dictionaries. In *HLT-NAACL*, pages 41–51.

Gary King and Will Lowe. 2003. An automated information extraction tool for international conflict data with performance as good as human coders: A rare events evaluation design. *International Organization*, 57(03):617–642.

Sebastian Krause, Enrique Alfonseca, Katja Filippova, and Daniele Pighin. 2015. Idest: Learning a distributed representation for event patterns. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL'15)*, pages 1140–1149.

Hanspeter Kriesi, Edgar Grande, Martin Dolezal, Marc Helbling, Dominic Höglinger, Swen Hutter, and Bruno Wüest. 2012. *Political conflict in western Europe*. Cambridge University Press.

Omer Levy, Yoav Goldberg, and Ido Dagan. 2015. Improving distributional similarity with lessons learned from word embeddings. *Transactions of the Association for Computational Linguistics*, 3:211–225.

Peter Makarov, Jasmine Lorenzini, and Hanspeter Kriesi. 2016. Constructing an annotated corpus for protest event mining. *NLP+CSS 2016*, page 102.

Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Diego Marcheggiani and Ivan Titov. 2017. Encoding sentences with graph convolutional networks for semantic role labeling. *EMNLP*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*, pages 3111–3119.

Peter F Nardulli, Scott L Althaus, and Matthew Hayes. 2015. A progressive supervised-learning approach to generating rich civil strife data. *Sociological Methodology*.

Clayton Norris, Philip Schrodt, and John Beieler. 2017. PETRARCH2: Another event coding program. *The Journal of Open Source Software*, 2(9), jan.

Sean P O'Brien. 2010. Crisis early warning and decision support: Contemporary approaches and thoughts on future research. *International Studies Review*, 12(1):87–104.

Brendan O'Connor, Brandon M Stewart, and Noah A Smith. 2013. Learning to extract international relations from political context. In *ACL (1)*, pages 1094–1104.

Robert Parker, Linguistic Data Consortium, et al. 2009. *English Gigaword Fourth Edition LDC2009T13*. Linguistic Data Consortium.

Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global vectors for word representation. In *EMNLP*, volume 14, pages 1532–1543.

Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 675–682. Association for Computational Linguistics.

Ellen Riloff and Rosie Jones. 1999. Learning dictionaries for information extraction by multi-level bootstrapping. In *AAAI/IAAI*, pages 474–479.

Michael Roth and Mirella Lapata. 2016. Neural semantic role labeling with dependency path embeddings. *arXiv preprint arXiv:1605.07515*.

Philip A Schrodt. 2006. Twenty years of the Kansas event data system project. *The political methodologist*, 14(1):2–8.

Philip A Schrodt. 2012. CAMEO: Conflict and mediation event observations event and actor codebook. Version: 1.1b3.

Mark Stevenson and Mark A Greenwood. 2005. A semantic approach to IE pattern induction. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, pages 379–386. Association for Computational Linguistics.

Mark Stevenson and Mark A Greenwood. 2006. Comparing information extraction pattern models. In *Proceedings of the Workshop on Information Extraction Beyond The Document*, pages 12–19. Association for Computational Linguistics.

Kiyoshi Sudo, Satoshi Sekine, and Ralph Grishman. 2003. An improved extraction pattern representation model for automatic IE pattern acquisition. *ACL*, pages 224–231.

Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2007. Extracting semantic orientations of phrases from dictionary. In *HLT-NAACL*, volume 2007, pages 292–299.

Leonid Velikovich, Sasha Blair-Goldensohn, Kerry Hannan, and Ryan McDonald. 2010. The viability of web-derived polarity lexicons. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 777–785. Association for Computational Linguistics.

Roman Yangarber, Ralph Grishman, Pasi Tapanainen, and Silja Huttunen. 2000. Automatic acquisition of domain knowledge for information extraction. In *Proceedings of the 18th conference on Computational linguistics-Volume 2*, pages 940–946. Association for Computational Linguistics.

Dengyong Zhou, Olivier Bousquet, Thomas Navin Lal, Jason Weston, and Bernhard Schölkopf. 2004. Learning with local and global consistency. In *NIPS*, volume 16, pages 321–328.