

# Automated Dictionary Generation for Political Event Coding \*

BENJAMIN J. RADFORD

*Event data provide high-resolution and high-volume information about political events and have supported a variety of research efforts across fields within and beyond political science. While these datasets are machine coded from vast amounts of raw text input, the necessary dictionaries require substantial prior knowledge and human effort to produce and update, effectively limiting the application of automated event-coding solutions to those domains for which dictionaries already exist. I introduce a novel method for generating dictionaries appropriate for event coding given only a small sample dictionary. This technique leverages recent advances in natural language processing and machine learning to reduce the prior knowledge and researcher-hours required to go from defining a new domain-of-interest to producing structured event data that describe that domain. I evaluate the method via actor-country classification and demonstrate the method's ability to generalize to new domains with the production of a novel event dataset on cybersecurity.*

\*Benjamin J. Radford received his Ph.D. in political science from Duke University (benjamin.radford@gmail.com). The author thanks Michael D. Ward, Scott De Marchi, Kyle Beardsley, and Mark J.C. Crescenzi for their support and feedback on drafts of this work.

## INTRODUCTION

Event data provide high-resolution and high-volume information about political events. Event datasets can be coded either by hand or with the aid of software, a process referred to here as “automated event coding.” While automated event coding promises reproducible, timely, and exhaustive data, several outstanding challenges limit its practical use to a subset of problems of interest for social scientists. Among these challenges is dictionary generation. Current automated event coding solutions require large dictionaries of actors, events, and event characteristics to be populated a priori such that pattern matching can be used to identify those dictionary entries in the raw text of news stories from which event data will be generated. The dictionaries are hand-coded and therefore suffer from many of the same limitations that hand-coded event datasets suffer from: they are costly to produce, require frequent updates, are not reproducible, and are vulnerable to the forgetfulness or oversight of human coders. This paper presents a novel method for generating dictionaries for event coding that ameliorates these problems. Automated dictionary generation (ADG) promises to allow researchers to rapidly generate novel datasets tailored to their research questions rather than adapting their research questions to fit existing event datasets.<sup>1</sup> By lowering the costs of dictionary generation, researchers will be better able to adapt existing event coding software to new domains and to iterate rapidly on their datasets.

The ADG method presented herein is demonstrated via two applications. The first application is the generation of a novel event dataset in a new domain, cybersecurity,

<sup>1</sup>While I will refer to this technique as automated, it might be better described as computer-assisted given that a minimal amount of human input is required at the outset in the form of seed terms or phrases.

for which appropriate event-coding dictionaries do not exist. This application consists of the generation and updating of verb, actor, agent, issue, and synset dictionaries. It is shown that ADG enables the expansion of automated event coding to new domains, and therefore new problem sets, with a minimal amount of researcher effort. The second application is the country-level classification of actors from existing CAMEO dictionaries. This demonstrates the effectiveness of the method in both generating and updating actor dictionaries in unsupervised and supervised settings. Performance is evaluated against existing “ground truth” data.

This paper proceeds by first discussing existing methods for producing event data in political science. Next the ADG method itself is detailed. The paper then offers two example applications of this method and introduces an event dataset on cybersecurity: CYLICON, the CYber LexICON event dataset. The paper concludes with a brief discussion of directions for future research in automated event coding.

## EVENT DATA IN POLITICAL SCIENCE

Political event data are produced both by hand and via automated processes. Most datasets of political events are still coded manually. This process is costly, time consuming, and irreproducible. However, hand-coded event data is popular due to the perceived control it affords researchers in leveraging their expertise to code events precisely. Hand coding also allows researchers to collect information from multiple sources to construct event records with details that may not be available from any single source. Notable hand-coded event datasets include the Armed Conflict Location and Event Dataset, the International Crisis Behavior dataset, the Militarized Interstate Dispute dataset, and the Conflict and Peace Databank (Raleigh et al. 2010; Brecher and Wilkenfeld 2000; Brecher et al. 2016;

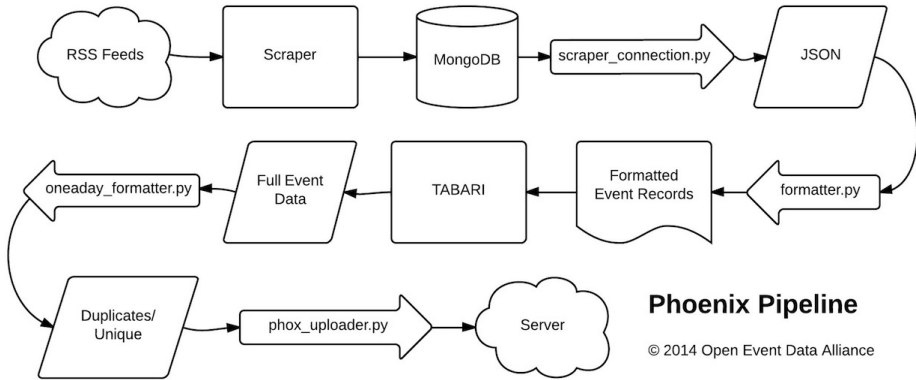


Figure 1. The Phoenix pipeline (Open Event Data Alliance 2015c).

Palmer et al. 2015; Azar 1980).

Since the mid 1990s, automated coding efforts for event datasets have grown in popularity (Schrodt 1998, 2011; Ward et al. 2013; Boschee et al. 2015; Caerus Associates 2015; Schrodt and Brackle 2013). In just the past several years, several event datasets have been introduced in political science: The Global Database of Events, Language, and Tone (GDELT), the Integrated Crisis Early Warning System (ICEWS) dataset, the Open Event Data Alliance’s Phoenix dataset (Leetaru and Schrodt 2013; Lustick et al. 2015; Open Event Data Alliance 2015b), the Cline Center’s Historical Phoenix Dataset (Althaus et al. 2017). These datasets provide information on individual events, usually at the daily level, with specific details about the actors involved. They also often provide geographic information at a subnational level. These datasets are enormous, typically comprising millions of events.<sup>2</sup>

<sup>2</sup>GDELT, for instance, claims 103 million events as of February, 2016 (The GDELT Project 2016). ICEWS comprises nearly 15 million events. For a brief discussion of the validity of these datasets, see Wang et al. (2016).

The event datasets listed above are built from streams of open-source news stories. The stories are processed through software that uses pre-defined dictionaries to infer the actors and actions they describe. Common software packages for this purpose include TABARI (Textual Analysis by Augmented Replacement Instructions) and PETRARCH (Python Engine for Text Resolution And Related Coding Hierarchy), both of which are successors to KEDS (Kansas Event Data System) (Schrodt 2011; Open Event Data Alliance 2015a; Schrodt 1998).<sup>3</sup> The Open Event Data Alliance, authors of PETRARCH, provide Figure 1 to illustrate their event-coding process. Raw stories are first collected from online sources. These are uploaded to a database and formatted to the specifications required by TABARI (or PETRARCH). The stories are then passed to TABARI (or PETRARCH) which uses the supplied dictionaries to produce structured data. The data are then de-duplicated using a one-a-day filter to remove multiple identically-coded event records from the same day. The resulting data are then uploaded to a server for distribution. Under ideal circumstances, human interaction is only required to select appropriate news sources, devise an ontology for the resulting structured data, and to populate the necessary dictionaries. However, this last step, dictionary creation, requires a substantial level of effort. The CAMEO verb dictionary used by PETRARCH and the Phoenix dataset is nearly 15,000 lines long and includes very specific phrases that would not necessarily be apparent to researchers a priori.<sup>4</sup> The country actors dictionary, just one of multiple actor dictionaries utilized by Phoenix, is nearly 55,000 lines long. As of 2014, the ICEWS actor dictionary was

<sup>3</sup>PETRARCH here refers to the original event-coding software to go by that name (Open Event Data Alliance 2015a). There are two additional event-coding software packages to go by the name PETRARCH: PETRARCH2 and Universal Dependency PETRARCH (Norris, Schrodt, and Beiler 2017; Open Event Data Alliance 2018).

<sup>4</sup>CAMEO, Conflict and Mediation Event Observations, is a common framework for event data and the basis for the three automated event datasets cited here (Schrodt, Gerner, and Yilmaz 2009).

over 102,000 lines long. Furthermore, as the relevant actors and language evolve, these dictionaries require regular updates to maintain up-to-date event data. Excerpts from the verb, country-actors, and synset dictionaries provided with PETRARCH are given in Table 1.

The purpose of event-coding dictionaries, like those used by TABARI and PETRARCH, is to provide an exhaustive list of the terms and phrases that map to a set of labels. In a fully automated event-coding solution, both the ontology and the dictionary could be produced without human intervention. The effort described here, however, focuses on the latter challenge: automating the process of synonym and near-synonym extraction and classification given a known ontology.

PETRARCH’s dictionary structure includes a verb dictionary, three distinct actor dictionaries, an agents dictionary, an issues dictionary, and a discard dictionary. The verb dictionary categorizes verb phrases into the sets of predetermined actions described by event data. The three actor dictionaries categorize persons and named organizations by their affiliations (i.e. country, organization type) and their roles related to the domain of interest. These dictionaries also resolve multiple spellings or representations of an entity’s name into a single canonical representation. The default PETRARCH coding scheme provides three actor dictionaries: country-affiliated actors, international actors, and non-state military actors. The agents dictionary describes how to classify unnamed entities. For example, the agents dictionary maps “thief” and “trafficker” to *criminal*. The issues dictionary identifies phrases common to the domain-of-interest to label news by topic. For example, the current Phoenix issues dictionary tags issues like *foreign aid*, *retaliation*, and *security services*. Finally, the discard dictionary identifies phrases that disqualify sentences or stories from being coded entirely. This helps to remove stories that might otherwise be erroneously coded. For example, sports reporting is omitted as

TABLE 1 Excerpts from dictionaries supplied with PETRARCH.

CAMEO Verbs	Synsets	Country-Actors
--- ABANDON [080] ---	&STRONGHOLD	JOHN_FOSTER_DULLES_
ABANDON	+STRONGHOLD	[USAEI 19060101-530121]
- SAID + MUST * POLICY [100]	+BASTION	[USAGOV 530121-590422]
- * HEADQUARTERS [0874]	+CITADEL	CHRISTIAN_A._HERTER_
- * OUTPOST IN + [0874]	+BLOCKHOUSE	[USAEI 19130101-590422]
:	:	:
---	---	---
WISH [---] ---	&CEASEFIRE	HAMID_KARZAI_
WISH	+TRUCE	+KARZAI_
- * + RECOVERY [018]	+ARMISTICE	+PRESIDENT_KARZAI_

it often uses the language of warfare to describe “victories,” “defeats,” and teams being “destroyed.”

The common CAMEO coding scheme is not a comprehensive description of public interactions between politically-relevant actors and agents. For researchers interested in types of interaction that do not conform to the existing dictionary structure, creation of new dictionaries is a necessary but costly step. The Phoenix verb dictionary contains many thousands of verbs and phrases parsed according to a particular format and organized within a predetermined ontology. Currently, not only must researchers do this parsing and organization by hand, but they must also begin with a comprehensive list of verbs and phrases that will comprise the dictionary. Historically, the work of identifying verb phrases and classifying them has been done by undergraduate or graduate research assistants. This is time consuming, expensive, and difficult to reproduce. The coding decisions made by research assistants are supposed to follow prescribed rules, but their actual judgments are not auditable. Tools adapted from machine learning and natural language processing can be leveraged to ameliorate these challenges of event data generation. The technique presented here relies primarily on a word embedding model called *word2vec*.

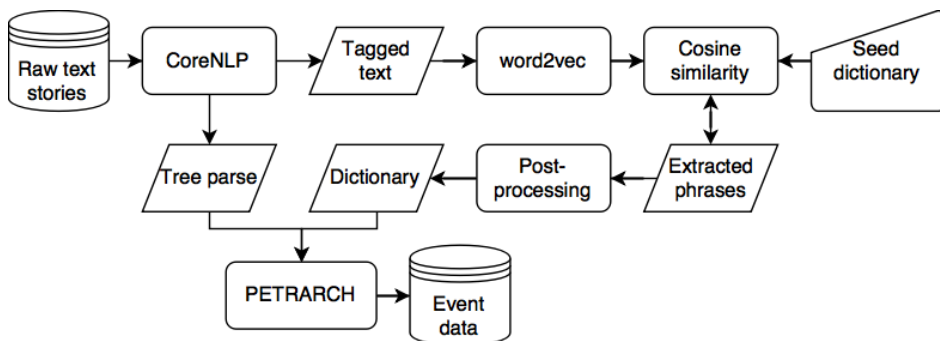


Figure 2. ADG pipeline.

## A METHOD FOR AUTOMATED DICTIONARY GENERATION

The ADG process consists of four steps. (1) First, techniques common to NLP tasks are used to pre-process the text corpus that is to be event-coded. This is a necessary step for both event coding by PETRARCH as well as the dictionary creation process. (2) Word2vec, a neural network language model (NNLM), is then used to learn a vector-space representation of the entire vocabulary. (3) Seed words and phrases, chosen according to a pre-defined ontology, are used to extract synonymous and nearly-synonymous words and phrases from the word2vec model that will populate the dictionaries. (4) Finally, a set of post-processing heuristics are applied to prune and format the dictionaries. While this entire process consists of multiple steps, the researcher is responsible only for supplying an ontology in the form of a small set of seed words and phrases. The process is diagrammed in Figure 2 and described in detail below. While the examples provided are drawn from the application of ADG to cybersecurity, the process is domain agnostic and can be applied widely to a variety of event domains.

*Step 1: Pre-processing*

Every story in the corpus that is to be event-coded is parsed and part-of-speech tagged using a shift-reduce parser, the fastest parser available from Stanford’s CoreNLP (Bauer 2014).<sup>5</sup> Additionally, CoreNLP’s named entity recognizer (NER) is used to tag named entities as one of *time*, *location*, *organization*, *person*, *money*, *percent*, and *date* (Finkel, Grenager, and Manning 2005).

Once the corpus has been parsed and named entities have been identified, two versions of the annotated text are saved. The first version is a representation of each sentence’s parse tree to be input into PETRARCH. The second version of the annotated corpus is formed by appending to each word both its entity-type tag and its part-of-speech tag. For example, the word “hackers” is transformed into “hackers:O:NNS” where “O” indicates that this word is not a named entity and “NNS” indicates a plural noun. “Snowden:PERSON:NNP” indicates that “Snowden” refers to a person and is a singular proper noun.<sup>6</sup> POS and NER-tagging each word and phrase in the corpus is necessary to retain sufficient information about each term to post-process the resulting dictionary entries.

The NER and POS-tagged corpus is then processed to produce multi-word phrases. The method chosen here for deriving phrases from the corpus is recommended by Mikolov, Sutskever, et al. (2013) and implemented in Rehurek and Sojka (2010). A robust literature on phrase detection exists but is out of scope for review here.<sup>7</sup> Candidate bigrams (two-word phrases) are scored according to their frequency relative to the frequency of the

<sup>5</sup>The shift-reduce parser is chosen only for its speed and so other parsers may be substituted here as necessary.

<sup>6</sup>For more on the Penn Treebank POS tags, see Santorini (1990).

<sup>7</sup>For more, please see Dunning (1993).

constituent words being found independently:

$$score(w_1, w_2) = \frac{count(w_1, w_2) - \delta}{count(w_1) \times count(w_2)} \quad (1)$$

The words  $w_1$  and  $w_2$  are concatenated into a single multi-word term,  $w_1\_w_2$ , if  $score(w_1, w_2)$  surpasses a pre-defined threshold.  $\delta$  is a discount factor that prevents spurious phrases from being formed by infrequently-occurring words. In order to produce phrases consisting of more than just two words, this algorithm is run iteratively. An example of this pre-processing is given in Figure 3.

### *Step 2: Vocabulary Modeling*

Once the text data have been tagged and phrases have been formed, a model is required to identify terms and phrases that are synonymous with the seed phrases. Word2vec is chosen for this purpose. The word2vec model is a single-hidden-layer, fully-connected, feed-forward neural network that has been shown to learn the meanings of words given their contexts in natural language texts. Word2vec produces word vectors, in the form of real-valued vectors, from raw text input in a process called embedding (Mikolov, Chen, et al. 2013; Rehurek and Sojka 2010). These word vectors are low-dimensional numeric representations of a vocabulary that preserve the syntactic and semantic relationships between words. Word2vec learns the meaning of words from the contexts in which they are found in the text. The importance of a word’s context is found in the distributional hypothesis, an assumption required by word2vec. Harris (1954), in describing the distributional hypothesis, explains that words more similar in meaning will occur among more similar contexts than will words that are dissimilar in meaning. Rubenstein and Goodenough (1965) demonstrate that “there is a positive relationship between the degree

These websites could contain specially crafted content that could exploit this vulnerability in Internet Explorer.

⇓

These:0:DT websites:0:NNS could:0:MD contain:0:VB  
specially:0:RB\_crafted:0:VBN content:0:NN that:0:WDT could:0:MD  
exploit:0:VB this:0:DT vulnerability:0:NN in:0:IN  
Internet:MISC:NN\_Explorer:MISC:NNP.

Figure 3. Example of pre-processing.

of synonymy (semantic similarity) existing between a pair of words and the degree to which their contexts are similar.”

Word2vec is actually a family of models that includes both a skipgram-based variant and a continuous bag of words (CBOW) variant.<sup>8</sup> The skipgram model takes as input a one-hot-encoded (dummy variable) vector of length  $V$ , where  $V$  is the size of the vocabulary, in which all values are 0 except for the target word,  $w_i$ , which is coded 1. The skipgram model then attempts to predict the context words that are most likely to be found adjacent to the target word. Context words,  $\{w_{i-k}, \dots, w_{i-1}, w_{i+1}, \dots, w_{i+k}\}$ , are those words that fall within a window of size  $k$  on either side of the target word.<sup>9</sup> The

<sup>8</sup>Word2vec builds on previous research into machine learning for natural language modeling, techniques for which include Latent Semantic Analysis and a variety of other NNLMs. Recently, Dhillon, Foster, and Ungar (2015) use singular value decomposition of a word-adjacency matrix to produce word embeddings. An extension of word2vec, called paragraph2vec (or doc2vec), estimates vector representations of groups of words in addition to the words themselves. These “documents” can be full sentences, paragraphs, or larger articles (Le and Mikolov 2014). Similar embedding models have explored character-level embedding and embedding based on global word co-occurrence counts (Bojanowski et al. 2016; Pennington, Socher, and Manning 2014).

<sup>9</sup>The window is randomly sampled from between 1 and  $k$  such that words further from the target word are, on average, weighted less heavily than words immediately adjacent to the target word. Note that  $k$  is a researcher-specified hyperparameter while  $V$  is the overall size of the vocabulary; typically  $V > k$ .

skipgram model therefore estimates a function,  $f(w_i)$ , that maps target word  $w_i$  to its likely context words,  $\{w_{i-k}, \dots, w_{i-1}, w_{i+1}, \dots, w_{i+k}\}$ . The output of the skipgram model is a softmax-normalized vector of length  $V$  where elements represent the probabilities that each corresponding word will appear in the context window of the input word.<sup>10</sup> The CBOW variant is the reverse of the skipgram model and predicts a target word given its context. Both CBOW and skipgram models can be estimated with any of several software packages including the one used here, *gensim* (Rehurek and Sojka 2010).

Word2vec consists of two weights matrices: an input weights matrix and an output weights matrix. By multiplying the input vector (shape  $1 \times V$ ) with the input weights matrix (shape  $V \times D$ ), a  $D$ -dimensional vector representation of the input word, its word vector, is formed. This vector representation is then multiplied by the output weights matrix (shape  $D \times V$ ) to produce the model’s output layer.<sup>11</sup> The softmax function (“activation”) is applied to this output layer. Because  $D \ll V$ , the hidden layer compresses the sparse input vectors into relatively small, dense vectors.<sup>12</sup> These word vectors are of interest because they encode semantic and syntactic relationships between words and can be used to measure word similarities. Furthermore, algebraic operations on this vector space produce intuitive results. The canonical example of this is the analogy task, often

<sup>10</sup>The softmax function, a multiclass generalization of the logistic function, is defined as  $\sigma(x_j) = \frac{\exp(x_j)}{\sum_{k=1}^K \exp(x_k)}$ . The softmax function maps a real-valued vector to a vector of values between zero and one that sums to one. It is therefore used to represent a probability distribution over discrete outcomes; in word2vec’s skipgram case, those outcomes are context words.

<sup>11</sup>Mikolov, Chen, et al. (2013) introduce NEG, a negative sampling objective function, for optimizing word2vec. For a discussion of the skipgram negative sampling word2vec objective function, see Goldberg and Levy (2014). Word2vec can be optimized via stochastic gradient descent as described by Mikolov, Chen, et al. (2013).

<sup>12</sup> $D$  is a parameter supplied by the researcher. Common values are 100 and 300.

demonstrated by showing that:

$$\overrightarrow{\text{king}} + \overrightarrow{\text{woman}} - \overrightarrow{\text{man}} \approx \overrightarrow{\text{queen}} \quad (2)$$

By adding the vector representation of “king” to the vector representation of “woman” and subtracting the vector representation of “man,” a well-trained word2vec model will produce a vector very near to the vector representation of “queen” (i.e. king:man::queen:woman).<sup>13</sup> Why word vectors exhibit these linear relationships is the subject of active research (Pennington, Socher, and Manning 2014; Arora et al. 2016).<sup>14</sup> English word embedding models are typically evaluated with a standard set of analogies like that offered by Mikolov, Chen, et al. (2013) to test a model’s ability to represent 14 categories of semantic and syntactic relationships.

<sup>13</sup>Note that the overhead arrow notation is here used to indicate a vector. For example:  $\overrightarrow{\text{king}} \in \mathbb{R}^n$ .

<sup>14</sup>Word embedding models are imperfect approximations of language; failure cases may include instances where antonyms share very similar word vector representations because they occur in similar contexts (Nguyen, Walde, and Vu 2016). Additionally, infrequent words and phrases tend not to be represented as well by word2vec as frequently-occurring words and phrases. In fact, implementations of word2vec accept a minimum count parameter to filter out infrequent words. Readers interested in an alternative approach to keyword discovery that does not rely on word embedding should consult King, Lam, and Roberts (2017). These issues will impact the performance of ADG. For example, infrequently-referenced actors may fall below the minimum count threshold chosen for word2vec and therefore not appear in the final dictionaries. Those infrequently-referenced actors that make the cut-off might still not occur frequently enough to produce reliable word vectors.

*Step 3: Term and Phrase Extraction*

Learning the corpus with word2vec allows us to easily identify synonyms or near-synonyms of our seed words and phrases. Given a seed phrase, a string search is performed on the model's vocabulary and all words and phrases that contain the given seed word or phrase are selected. The word vectors associated with the resulting words and phrases are retrieved. These vectors are element-wise averaged to produce a single category-wide vector. The element-wise average is taken as  $\|\sum_{\vec{w} \in C_i} \vec{w}\|_2$  where  $\sum_{\vec{w} \in C_i} \vec{w}$  is the element-wise sum of all word vectors,  $\vec{w}$ , in category  $C_i$ . The resulting vector is  $l^2$  normalized.<sup>15</sup> Then, the top  $n_i$  most similar terms and phrases to each mean category vector are extracted from the word2vec model. Similar words and phrases are identified by first computing the cosine similarities of all word vectors with the category mean vector. Cosine similarity, defined as  $(\vec{x} \cdot \vec{y}) / (\|\vec{x}\| \times \|\vec{y}\|)$ , is a measure of the angle between two vectors and is particularly useful for comparing high-dimensional vectors. Cosine similarity is used to rank-ordered all terms and phrases in the word2vec model's vocabulary by their similarity to the mean category vector in descending order. The top  $s$  most similar terms and phrases are chosen as candidates to populate the relevant category in the event-coding dictionary.<sup>16</sup>

<sup>15</sup>The  $l^2$  norm for vector  $\langle x_1, x_2, \dots, x_n \rangle$  is given by  $\sqrt{\sum_{i=1}^n x_i^2}$ .

<sup>16</sup> $s$  is a researcher-selected value that puts an upper limit on the number of terms and phrases that will constitute the dictionary. However, automated post-processing steps described in the appendix may result in the inclusion of fewer terms and phrases. Values for  $s$  used here are 300, 300, 50, and 25 for the verb, actor/agent, synset, and issue dictionaries, respectively.

*Step 4: Post-processing*

Extracted terms are then post-processed according to a set of rules associated with the dictionary they are meant to comprise. These post-processing steps can be automated. The set of post-processing rules can be found in the online Appendix. The post-processing is necessary to coerce the extracted terms and phrases into the dictionary formats expected by PETRARCH. This involves, among other things, grouping verb phrases by their common verbs and tagging each dictionary entry with a category tag. A post-processing filter that removes phrases from the verb dictionary if they do not include at least one verb is also applied.

ADG represents a major step towards fully-automated event-data coding for novel domains. Because this process can be done largely without human interaction and the content of the dictionaries are a function of the raw data that are to be event-coded, the dictionaries can be updated in tandem with the event dataset itself; new verb phrases, actors, or agents can be learned by the underlying models as they enter the relevant domain's vocabulary. Additionally, because the process described herein relies on only a small amount of initial researcher input data and the raw text data itself, the process of event data generation is made more fully reproducible from start to finish.

**CYLICON: A CYBER EVENT DATASET**

This method of ADG for event coding is now applied to a novel domain for event data: cybersecurity. First, a cybersecurity ontology is selected and seed phrases are chosen to represent each category of that ontology. Five dictionaries are generated: verbs, actors,

TABLE 2 *Verb dictionary seeds*

Category	Seed Phrase
DEFACED	DEFACED:O:VBD
PATCHED	PATCHED:O:VBD
INFILTRATED	BREACHED:O:VBD
LEAKED	LEAKED:O:VBD
PHISHED	PHISHED:O:VBD
DDOS	DISTRIBUTED:O:VBN_DENIAL-OF-S . . .
INFECTED	INFECTED:O:VBD
VULNERABILITY	VULNERABILITY:O:NN
ARRESTED	ARRESTED:O:VBD
CENSORED	CENSORED:O:VBD

agents, synsets, and issues.<sup>17</sup> Only one seed phrase is provided per category. Seed phrases are shown in Tables 3 and 2 and in the appendix. For each seed term or phrase, the average vector of all terms and phrases containing the seed is computed and similar terms and phrases are identified according to the described ADG procedure. The extracted candidate terms and phrases are then post-processed and formatted into PETRARCH-styled dictionaries; no manual changes have been made to the dictionaries at any point after the input of the 26 seed phrases (one per category).

Ten categories of events are identified for the verb dictionary: defacements, DDOS events, infiltrations, leaks, infections, vulnerability discoveries, arrests, patches, phishing attacks, and censorship incidents.<sup>18</sup> The ten seed phrases are representative examples of

<sup>17</sup>The word2vec model was trained on a convenience sample of cybersecurity news data from a number of sources. The model is trained according to the *gensim* default parameters except `min_count` 10, `window` 10, and `vector size` 300.

<sup>18</sup>These categories were chosen for exploratory purposes. For future iterations of CYLICON, existing cybersecurity ontologies, to include those developed by Herzog, Shahmehri, and Duma (2007) and Swimmer (2008), will be considered as alternatives. These will require technical reports of cybersecurity events as opposed to the newswire-like corpus used here. They may also require adjustment to fit social science applications rather than their intended audience of cybersecurity experts and incident responders.

TABLE 3 *Actor & agent dictionary seeds*

Category	Seed Phrase
HACKER	HACKER:O:NN
RESEARCHER	RESEARCHER:O:NN
WHISTLEBLOWER	WHISTLEBLOWER:O:NN
USERS	USERS:O:NNS
ANTIVIRUS	ANTIVIRUS:O:NN

verb phrases for each category.<sup>19</sup> These are chosen by the researcher. The extracted verb dictionary contains 640 verbs and phrases after de-duplication and post-processing. The number of extracted phrases is due, in large part, to the a minimum similarity threshold that is set by the researcher; terms and phrases must surpass this threshold with respect to the average category vector in order to be included in the final dictionaries. Here, a minimum cosine similarity of 0.6 has been chosen.

The new categories of actors and agents introduced in CYLICON include hackers, researchers, users, whistleblowers, and antivirus companies/organizations. These categories are appended to the existing actor and agent classifications already found in the default Phoenix dictionaries. New issue categories are appended to the issues already supplied with PETRARCH and include TOR, 0Day, hacktivism, DDOS, social engineering, and state-sponsorship. Synsets are produced for categories including hardware, virus, web asset, software, and computer.<sup>20</sup>

The selected text corpus represents a convenience sample of 77,410 documents collected from online sources including cybersecurity-related blogs and news sites. Roughly 22,000 articles are sourced from the news section of [www.softpedia.com](http://www.softpedia.com). The

<sup>19</sup>In fact, some seed terms are nouns rather than verb phrases. For example, the seed word for the vulnerability discovery category is simply `VULNERABILITY:O:NN`.

<sup>20</sup>These categories are not mutually exclusive.

remaining stories are largely sourced from blogs and technology-oriented news sites, the largest of which include feed aggregators, theregister.com, csoonline.com, circleid.com, and darkreading.com. There are 1,231 unique sources represented in the corpus. These sources are not a representative sample of cybersecurity events and were instead selected due to their relatively high concentration of relevant cybersecurity event stories. Collection occurred during 2014 and the latter part of 2015 and was inconsistent over time due to heterogeneity among sources with respect to the availability of archival text.

CYLICON includes 671 events in total. Arrests make up the largest category with 211 events, followed by infiltration (200), leaks (97), defacements (97), patches (19), infections (19), DDOS attacks (17), vulnerability discoveries (5), phishing attacks (5), and censorship incidents (1). Infiltration is a common category as many verb phrases from cybersecurity reporting accurately map to it. For example, phrases that include the words “breached” and “hacked” are often classified as infiltration by the ADG process. Additionally, when websites are defaced, it is common for reports to describe the websites as having been “breached and defaced,” indicating that the incident could be accurately assigned to either or both categories. Often, popular reporting on cybersecurity is not precise enough to distinguish the characteristic of a particular “hacking” event in a single sentence. Because of this, a bias towards infiltration coding is induced. If the coded sentence explains that a target was “hacked” and a second sentence explains that the event resulted in the defacement of the target’s website, PETRARCH will fail to connect the defacement to the hacking event and will therefore code the event as an infiltration rather than a defacement. The discovery of vulnerabilities, issuance of patches, and phishing attempts, while very common, often go unreported in the news sources utilized here. They also tend not to conform to the source-action-target triple expected by PETRARCH. Of the 640 verb phrases in the CYLICON dictionaries, 157 of them account for all of the

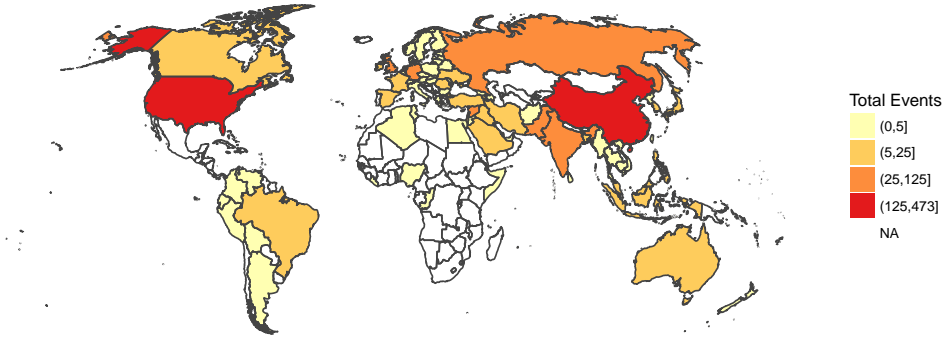


Figure 4. *Spatial distribution of actors in CYLICON.*

coded events. This is a fifteen-fold increase over the size of the verb seed dictionary.

The geographic distribution of actors involved in cyberspace according to CYLICON is shown in Figure 4. This map corresponds to conventional wisdom about the most active actors in cybersecurity-related events (The Economist 2012; Clapper 2015; Akamai, Q4, 2015). However, this map is not representative of the entire CYLICON dataset; not all relevant actors are geo-coded. Of 1,338 total coded actors, 1,245 are assigned to specific countries. PETRARCH attempts to assign country codes to actors and agents when they can be inferred from the text; for example, the phrase "Syrian hackers" may be coded as SYRHAC. Actors affiliated with international organization or otherwise unaffiliated with specific countries are, of course, not included in the map. Country associations for cybersecurity-based actors and agents have not been inferred for CYLICON.<sup>21</sup> The USA is the most prominent country in CYLICON with 473 events followed by China (145), Great Britain (60), India (43), Pakistan (38), and Russia (38). 82 unique countries are represented in total.

Because event data from PETRARCH are dyadic, we can also examine country pair

<sup>21</sup> Actors identified by the ADG process are assigned the country code XXX by default.

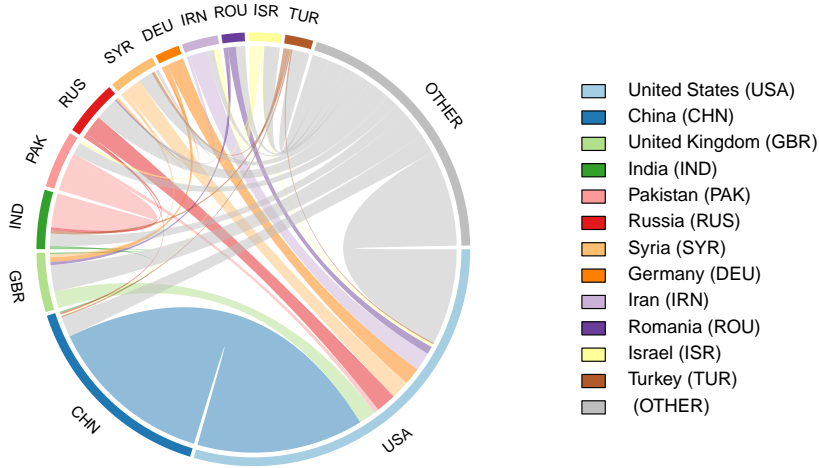


Figure 5. Top country dyads in CYLICON.

interactions. Figure 5 represents the most common dyadic pairs in CYLICON. Chord plots, common in network analysis applications, represent the volume of interaction between nodes or, in this case, countries. This particular chord plot is non-directed and does not include self-connections. The top 12 countries (by volume of events) are plotted and the remaining 70 are grouped into the category “other” for visual clarity. The larger edges conform to the expectations of Valeriano and Maness (2014); regional pairs and rivals are apparent in the graph. The United States is most active with China and Russia. India and Pakistan account for the majority of one another’s cyber events. Iran interacts primarily with the United States and Israel.

To better illustrate the successes and shortcomings of CYLICON, a selection of events are examined alongside their original text. Event codes are indicated by the triplet ACTOR1 ACTOR2 ACTION preceding each sentence. Selected sentences and their corresponding

data are enumerated in the list below, beginning with examples of accurate coding and ending with examples of inaccurate coding. Commentary follows.

1. ISR USAELIGOV INFILTRATED: “According to FBI, in the Year 2000 Israeli Mossad had penetrated secret communications throughout the Clinton administration, even Presidential phone lines.”<sup>22</sup>
2. USACOP EST ARRESTED: “After the Estonian masterminds were apprehended by the FBI, the DNSChanger Working Group was established and the cleaning process began.” (Kovacs 2012b)
3. MYS PHLGOVMEDHAC DDOS: “After Anonymous Malaysia launched distributed denial-of-service (DDOS) attacks against several Philippines government websites, Filipino hackers went on the offensive, defacing a large number of commercial websites.” (Kovacs 2013)
4. USA USAMIL INFECTED: “US officials did not provide details on the status of the ‘corrupt’ software installed on DoD computers, but common sense points us to believe it was removed back in 2013.” (Cimpanu 2015)
5. BGD MED BGD DEFACED: “A Bangladeshi publisher of secular books has been hacked to death in the capital Dhaka in the second attack of its kind on Saturday, police say.” (BBC 2015)
6. IRNGOVGOVMIL USA INFILTRATED: “Head of Iran’s Civil Defense Organization Gholam Reza Jalali told the agency that the country never hacked financial institutions from the United States.” (Kovacs 2012a)

<sup>22</sup>Source unavailable due to dead hyperlink.

The first four examples are all accurately coded by PETRARCH. Item 1 is correctly identified as an instance of infiltration and the actors are accurate if imprecise (PETRARCH codes Mossad as ISR rather than ISRSpy). In Item 2, the ADG process identified “were apprehended” as indicative of arrest. While Item 3 is correctly labeled a DDOS event, PETRARCH has mistakenly associated the term “hackers” with the target actor rather than with Anonymous Malaysia. Item 4 highlights the difficulty associated with coding infection events. The “corrupt software installed” indicates that a malware infection event has occurred. However, as is often the case with infection events, a source actor is not described. In this case the target actor is accurately identified but the source actor is coded as the United States, which is not supported by the given text. Note that none of the verb phrases in Items 1, 2, or 4 were included in the seed terms.

Items 5 and 6 were incorrectly coded. The incorrect coding in item 5 resulted from the dual meaning of the verb “hacked.” It is possible that with a larger ontology, one that includes both computer infiltration and murder, “hacked\_to\_death” would be accurately coded. However, without a method for automatically pruning erroneously-coded phrases from the dictionaries, edge cases like this must be identified and removed by hand. No

TABLE 4 *Accuracy by event category*

	INCORRECT	AMBIGUOUS	CORRECT
ARRESTED	45	2	164
CENSORED	0	0	1
DDOSSED	4	4	9
DEFACED	32	14	51
INFECTED	8	1	10
INFILTRATED	44	11	145
LEAKED	45	1	51
PATCHED	14	0	5
PHISHED	5	0	0
VULNERABILITY	4	0	1
TOTAL	201	33	437

manual pruning has been performed on these dictionaries and so edge cases remain. Item 6 is incorrectly coded because the sentence itself is a denial of the action that was identified. An Iranian official denies that his country had hacked into financial institutions in the United States but PETRARCH interpreted the sentence to mean that the event had, in fact, occurred.<sup>23</sup>

All CYLICON events have been reviewed manually and scored to help quantify the efficacy of automatically-generated event data dictionaries. The text content associated with each event is inspected and event codes are manually assigned without any knowledge of the CYLICON-assigned codes. In the case that multiple events are explicitly described (e.g. "...have breached and defaced..."), all appropriate events are assigned. When only one even is described (e.g. "...have defaced..."), only that specific event is assigned. When the language is ambiguous, all reasonable assignments are made but the event is also labeled as "ambiguous." Only the action or event type field is evaluated as only the verb dictionary was produced completely via ADG. The CYLICON actor, agent, and issue dictionaries are a combination of the Phoenix hand-coded dictionaries and automatically-generated dictionaries and are therefore not evaluated. Events are scored as correct if the associated action code from CYLICON is among the manually-identified event types for a given sentence. Events are scored as ambiguous if the associated action code from CYLICON is among the manually-identified event types but the text itself is ambiguous rather than explicit. For example, "Hackers have attacked servers..." is ambiguous because it could reasonably describe a DDOS event, an infiltration event, or a defacement. Events are considered incorrect if they fall into neither of the above two cases.

<sup>23</sup>The prevailing wisdom is that Iran was complicit in the attacks in question (Volz and Finkle, March 25, 2016). However, PETRARCH failed to code the event accurately given the supplied context.

Table 4 presents the results of this review by event category. Overall accuracy, the number of correctly-coded events and ambiguous events divided by the total number of coded events, is 70%. If ambiguous events are instead considered inaccurate, the accuracy of coded events falls to 65%. These values are in line with or above the reported human coder performance on top-level event categories. King and Lowe (2003) report that trained undergraduates can correctly classify events by their aggregate (top-level) event category between 39% and 62% of the time.<sup>24</sup> Schrodt and Brackle (2013) report machine-coding accuracy percentages for TABARI on the ICEWS project in the low- to mid-70s. The false positive rate, the percentage of sentences incorrectly determined by PETRARCH to contain any event, is 16%.<sup>25</sup> This performance is achieved despite requiring only minimal researcher-hours and one seed phrase per category.

#### ACTOR CLASSIFICATION

In the CYLICON application, ADG was shown to produce dictionaries capable of event-coding a corpus in a novel domain with minimal researcher input. Now I will demonstrate that an extension to the ADG process can be used to not only identify new terms and phrases, including actors, but to also associate actors with countries. It is assumed, for this task, that a list of actors and a list of countries are available to the researcher, possibly as the result of ADG, but that the associations between actors and countries are unknown. Actors are then classified by country using both unsupervised and supervised approaches. As with the ADG method presented above, word2vec is central to this task.

<sup>24</sup>The maximum accuracy value increases to 72% when event types are weighted by frequency.

<sup>25</sup>King and Lowe (2003) note that their event coding software, *Reader*, suffers from a higher false positive rate than the human coders.

TABLE 5 ICEWS data summary.

	Documents	Learned words and phrases
<i>ICEWS90</i>	570,488	682,324
<i>ICEWS183</i>	1,210,483	1,136,519
<i>ICEWS365</i>	2,441,345	1,793,776
<i>ICEWS730</i>	5,058,635	2,605,372

As with ADG, a word2vec model is first trained on a text corpus. In the fully unsupervised case, the actors' learned representations from the word2vec model are compared to the learned representations associated with locations (e.g. city, province, and country vectors). The k-nearest neighbors algorithm is then used to classify actors based on proximity to location vectors. In the supervised approach, it is assumed that the researcher has access to an existing country-actor dictionary that is in need of updating. A random forest is used to build a predictive model that maps actor vectors to predicted classes (i.e. country labels). Both methods are evaluated against a "ground truth" dataset.

To build training and test sets, the Phoenix country-actor dictionary is compared to the word2vec model vocabulary and those actors that are present in both the word2vec model and the dictionary are selected.<sup>26</sup> Because these actors are already classified by country in the Phoenix country-actor dictionary, a "ground truth" actor-country dataset can be produced for evaluation purposes. When actors are assigned multiple codes over time in the Phoenix dictionary, the first code, generally the earliest in time, is selected as that actor's country affiliation. It is uncommon for a single actor to be assigned to more than one country.

Word embedding models are sensitive to both corpus size and corpus pre-processing techniques. For this reason, multiple corpora are selected to train word2vec models and

<sup>26</sup>The Phoenix dictionaries are chosen over the ICEWS dictionaries for consistency with the rest of the paper.

the results are compared. A model provided by Google and trained on the entirety of Google News content is used to prove the viability of this method. Additionally, a series of models trained on news data from the ICEWS dataset are examined to test the feasibility of this method for researchers unable to obtain a corpus as thorough as Google's.<sup>27</sup>

Google's model, here referred to as *GoogleNews*, was trained on 100 billion words and resulted in a dataset describing 3 million unique words and phrases as vectors of length 300 (Google 2015). The ICEWS text corpus for 2013 and 2014 is used to train four models.<sup>28</sup> These models cover 90 days of news, 183 days of news, 365 days of news, and 730 days of news and are referred to as *ICEWS90*, *ICEWS183*, *ICEWS365*, and *ICEWS730*, respectively. Table 5 summarizes the ICEWS data. The ICEWS corpus consists of only those news stories that contain events from the ICEWS event dataset. ICEWS events are CAMEO-coded and therefore represent instances of political conflict and cooperation. ICEWS texts are drawn from roughly 300 different publishers (Raytheon BBN Technologies 2015).

For illustration purposes, the word vectors for actors in the ICEWS730 corpus are plotted in Figure 6. t-distributed Stochastic Neighbor Embedding (t-SNE) is used to project the word vectors into a two dimensional space while preserving local distances between neighboring points. t-SNE, a non-linear dimensionality reduction technique, was introduced by Maaten and Hinton (2009) and implemented for R by Radford (2017). All actors from the top ten countries (by number of unique actors) are colored by their associated country. The figure shows clustering by country and some separation between

<sup>27</sup>For the ICEWS-based models, the `min_count` parameter is set to 3 and the `vector size` parameter is set to 300. Other parameters are left at the default values.

<sup>28</sup>Unfortunately, the ICEWS corpus is only available to researchers involved with the ICEWS project.

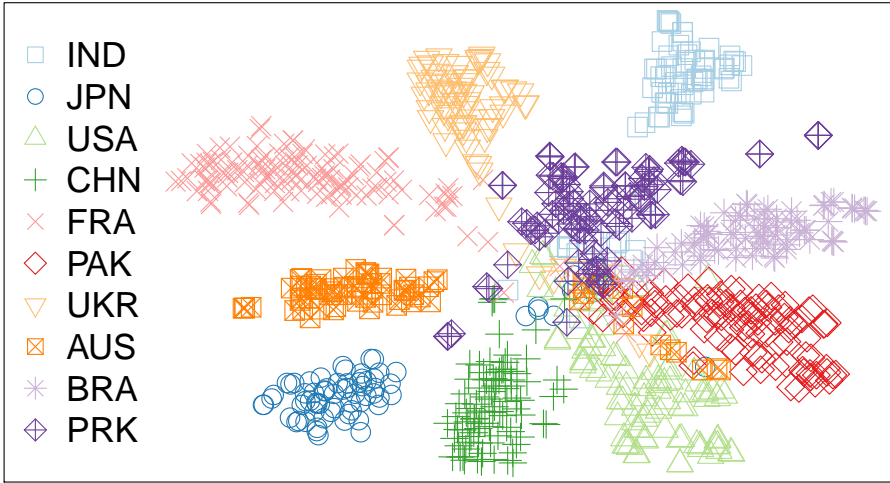


Figure 6. Embedding of actor vectors colored by ground truth country.

Note: t-SNE projection, ICEWS730 corpus.

the actors of different countries. Note that t-SNE is fully unsupervised and that any clustering by country observed in the plot is a function of the word vectors alone.

TABLE 6 Model performance on classifier task.

Model	Actors	Locations	Countries	Accuracy				
				k=1	k=3	k=5	k=7	RF
<i>ICEWS90</i>	2884	2271	182	0.25	0.27	0.29	0.30	0.26
<i>ICEWS183</i>	3890	2447	182	0.29	0.31	0.34	0.35	0.32
<i>ICEWS365</i>	4764	2630	182	0.32	0.34	0.37	0.38	0.35
<i>ICEWS730</i>	5096	2756	183	0.37	0.39	0.42	0.43	0.43
<i>GoogleNews</i>	4932	2616	183	0.58	0.62	0.64	0.64	0.54

### Unsupervised Actor Classification

In the first classification task, a set of location names and a set of actors to classify by country are given, but no training data are available. The objective is to simulate the process of ingesting news and producing country-actor dictionaries for event coding

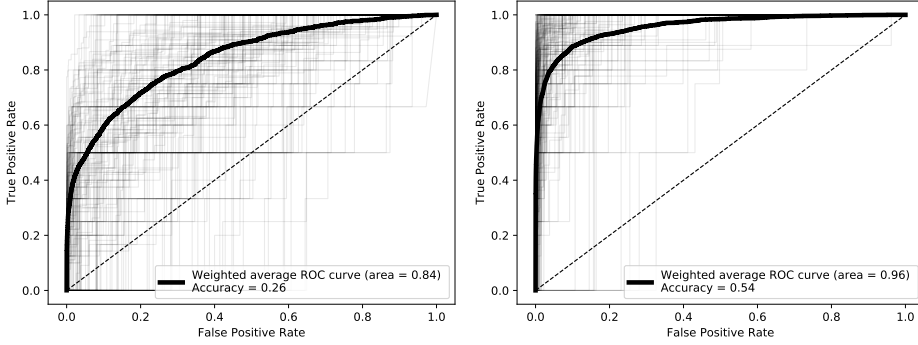
without human intervention. Actor names are assumed to have been extracted using ADG or NER while location names are assumed to be available from, for example, a gazetteer.

A cosine similarity matrix is constructed between the vectors associated with the actors' names and the vectors associated with location names. Each actor is then assigned to the country associated with the location whose vector is most similar, by cosine similarity, to the actor's vector. The process is tested for all five word2vec models and the results (test-set classification accuracy) are shown in Table 6. Additionally, k-nearest neighbors voting is applied for odd values of k between 3 and 7.

All five models substantially outperform weighted random assignment in classifying actors.<sup>29</sup> The worst performing model correctly classifies 25% of the actors. Models trained on larger corpora achieve better performance than those trained on smaller corpora and k-nearest neighbors voting improves accuracy as k increases. The model based on *GoogleNews* correctly classifies between 58% and 64% of all actors in the Phoenix country-actor dictionary. The next highest scores are achieved by *ICEWS730*, correctly classifying between 37% and 43% of actors. Interestingly, more Phoenix actors and locations are identified in the *ICEWS730* model's vocabulary than in *GoogleNews*'s. This is likely due in large part to the ability of the researcher to match data pre-processing procedures between the dictionary and the *ICEWS* models.<sup>30</sup> Previous work in this area has focused on event-level location prediction and is therefore not directly comparable to the actor-level location prediction results reported here. Lee, Liu, and Ward (2016) report up to 84% accuracy in the supervised task of predicting which location term in a news

<sup>29</sup>Simulations show that random assignment, weighted by class proportion, correctly classifies 1% of actors, on average.

<sup>30</sup>The discrepancy in performance between *GoogleNews* and *ICEWS730* could be the result of corpus size or model training parameters.



(a) ICEWS90 corpus.

(b) GoogleNews corpus.

Figure 7. Multiclass ROC plots for random forest classifier. Each thin line represents the ROC curve for a single country classifier evaluated in one-versus-all fashion. The bold line represents the weighted average ROC curve for all countries.

story describes the location in which an event occurred (as opposed to locations that are referenced for background or context). An unsupervised method for event-level location prediction, Mordecai, is introduced by Halterman (2017), but performance metrics are not yet available.

### Supervised Actor Classification

The supervised test assumes that the researcher has access to a training sample of classified terms or phrases. In this case, the set of actor-location pairs from the Phoenix country-actor dictionary is split in half to produce a training set and a test set. The split is performed via simple stratified random sampling such that the distribution of labels (countries) is matched across the training and test set. This has the effect of preventing situations in which the training set excludes entire categories. Actors from countries that appear only once in both the word2vec model and the country-actor dictionary are excluded as they cannot be represented in both the training and test sets. A random forest model is then

learned on the training set and used to predict labels for the held-out test set.<sup>31</sup> Actor word vectors constitute the input features of the random forest model that attempts to predict country affiliation.

Because it is possible to compute class membership probabilities from a the random forest model, receiver operating characteristic (ROC) plots can be drawn to visualize model performance. The worst- and best-case examples are depicted in Figures 7a and 7b. The accuracy of the random forest on each word2vec model is listed in the *RF* column of Table 6. Despite the use of ground truth labels, the accuracy of the random forest model largely mirrors the accuracy of the proposed unsupervised classification approach. The random forest successfully classifies 26% of actors on the smallest corpus and 54% of actors on the largest corpus, *GoogleNews*. The models' corresponding weighted AUC values, the areas under the ROC curves, are 0.84 and 0.96, respectively. Unlike accuracy, the AUC measures classification performance of a model across all possible thresholds; an AUC of 1 indicates perfect classification while an AUC of 0.5 is equivalent to random classification. High AUC values in the range seen here, above 0.8, are indicative of strong classification performance when averaged across all possible thresholds.<sup>32</sup>

## CONCLUSION

The ADG process described here allows researchers to quickly produce novel event datasets specific to their topics of interest. With minimal input from the researcher, ADG produces

<sup>31</sup>*scikit-learn* is used to learn the random forest model (Pedregosa et al. 2011). Parameters are left at their default values with the exceptions of `max_depth` 20 and `n_estimators` 1000.

<sup>32</sup>The threshold here being the minimum predicted probability required to assign an actor to a particular country.

dictionaries of pre-categorized words and phrases for use with existing event coding software. In a demonstration of its application, ADG was used to populate and update a set of dictionaries for coding events in an entirely new domain for event data—that of cybersecurity. An extension to the ADG method also shows some success in associating actors with their affiliated countries.

While ADG takes a substantial step in the direction of a fully-automated event coding solution, work remains to be done in this area. Event coding software itself, like PETRARCH, remains largely heuristic-based. The stacking of multiple analysis techniques for sentence parsing, phrase-extraction, and named entity recognition, among others, compounds errors that lead to sub-optimal event coding. Future efforts should leverage advances in machine learning to minimize the application of heuristics and stacking of text pre- and post-processing steps.<sup>33</sup>

End-to-end event coding models may, for instance, facilitate the customization of event datasets through transfer learning.<sup>34</sup> For example, a model may be trained to produce CAMEO-coded event data from news and then adapted, with the help of a relatively small training set, to produce cybersecurity event data instead. This would allow novel event datasets to be generated for user-specific purposes with only a small number of “gold standard” training samples. An extension to the ADG process presented here would replace the word2vec component with a bilingual embedding model like BilBOWA (Gouws, Bengio, and Corrado 2015). BilBOWA requires only a parallel bilingual corpus in order to align separate word embedding models in two different languages and could therefore be used in the ADG process to extract bilingual dictionaries.

ADG demonstrates that even unstructured text can be converted into structured data

<sup>33</sup>See IARPA (2018) for one such effort.

<sup>34</sup>For more on transfer learning, see Pan and Yang (2010).

suitable for social science inquiry with minimal researcher input. As machine learning and neural network-based models continue to advance the state-of-the-art in data analysis across fields, their application to the social sciences promises to similarly revolutionize how we measure, interpret, and understand political phenomena.

## REFERENCES

Akamai. Q4, 2015. “State of the Internet - Security.” 2, no. 4 (Q4).

Althaus, Scott L., Joseph Bajjalieh, John F. Carter, Buddy Peyton, and Dan A. Shalmon. 2017. “Cline Center Historical Event Data.” June 26. <https://www.clinecenter.illinois.edu/data/event/phoenix>.

Arora, Sanjeev, Yuanzhi Li, Yingyu Liang, Tengyu Ma, and Andrej Risteski. 2016. “RAND-WALK: A Latent Variable Model Approach to Word Embeddings.” *arXiv:1502.03520v7* (July 22).

Azar, Edward E. 1980. “The Conflict and Peace Data Bank (COPDAB) Project.” *The Journal of Conflict Resolution* 24, no. 1 (April).

Bauer, John. 2014. “Shift-Reduce Constituency Parser” [in English]. The Stanford Natural Language Processing Group. <http://nlp.stanford.edu/software/srparser.shtml>.

BBC. 2015. *Bangladeshi secular publisher hacked to death*. Online. <http://www.bbc.co.uk/news/world-asia-34688245>, October 31.

Bojanowski, Piotr, Edouard Grave, Armand Joulin, and Tomas Mikolov. 2016. “Enriching Word Vectors with Subword Information.” *arXiv preprint arXiv:1607.04606*.

- Boschee, Elizabeth, Jennifer Lautenschlager, Sean O'Brien, Steve Shellman, James Starz, and Michael Ward. 2015. "ICEWS Coded Event Data." <http://dx.doi.org/10.7910/DVN/28075> V15.
- Brecher, Michael, and Jonathan Wilkenfeld. 2000. *A Study of Crisis*. University of Michigan Press.
- Brecher, Michael, Jonathan Wilkenfeld, Kyle Beardsley, Patrick James, and David Quinn. 2016. "International Crisis Behavior Data Codebook, Version 11." <http://sites.duke.edu/icbdata/data-collections>.
- Caerus Associates. 2015. "Phoenix Event Data Set Codebook 0.0.1b." [https://s3.amazonaws.com/oeda/docs/phoenix\\_codebook.pdf](https://s3.amazonaws.com/oeda/docs/phoenix_codebook.pdf).
- Cimpanu, Catalin. 2015. "Military Contractors That Used Russian Programmers for DoD Software Get Fined by US Govt." <http://news.softpedia.com/news/military-contractors-that-used-russian-programmers-for-dod-software-get-fined-by-us-govt-495827.shtml>, *Softpedia Security News* (November 6).
- Clapper, James R. 2015. *Worldwide Threat Assessment of the US Intelligence Community*. [http://cdn.arstechnica.net/wp-content/uploads/2015/02/Clapper\\_02-26-15.pdf](http://cdn.arstechnica.net/wp-content/uploads/2015/02/Clapper_02-26-15.pdf). Senate Armed Services Committee, February 26, 2015.
- Dhillon, Paramveer S., Dean P. Foster, and Lyle H. Ungar. 2015. "Eigenwords: Spectral Word Embeddings." <http://www.pdhillon.com/dhillon15a.pdf>, *Journal of Machine Learning Research* 16.
- Dunning, Ted. 1993. "Accurate Methods for the Statistics of Surprise and Coincidence." *Computational Linguistics* 19 (1): 61–74.

- Finkel, Jenny Rose, Trond Grenager, and Christopher Manning. 2005. “Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling.” <http://nlp.stanford.edu/~protect/unhbox/voidb@x/penalty/@M/{manning/papers/gibbscrf3.pdf>, *Proceedings of the 43rd Annual Meeting of the Association of Computational Linguistics*: 363–370.
- Goldberg, Yoav, and Omer Levy. 2014. “word2vec Explained: Deriving Mikolov et al.’s Negative-Sampling Word-Embedding Method.” *arXiv* (February).
- Google. 2015. “word2vec.” <https://code.google.com/p/word2vec/>.
- Gouws, Stephan, Yoshua Bengio, and Greg Corrado. 2015. “BilBOWA: Fast Bilingual Distributed Representations without Word Alignments.” *Proceedings of the 32nd International Conference on Machine Learning* 37.
- Halterman, Andrew. 2017. “Mordecai: Full Text Geoparsing and Event Geocoding.” *The Journal of Open Source Software* 2 (9). doi:10.21105/joss.00091.
- Harris, Zellig S. 1954. “Distributional Structure.” *WORD* 10 (2-3): 146–162.
- Herzog, Almut, Nahid Shahmehri, and Claudiu Duma. 2007. “An Ontology of Information Security.” *International Journal of Information Security and Privacy* 1 (4): 1–23.
- IARPA. 2018. “DRAFT BROAD AGENCY ANNOUNCEMENT: Better Extraction from Text Towards Enhanced Retrieval (BETTER).” *IARPA-BAA-18-05*.
- King, Gary, Patrick Lam, and Margaret E. Roberts. 2017. “Computer-Assisted Keyword and Document Set Discovery from Unstructured Text.” *American Journal of Political Science* 61 (4): 971–988.

- King, Gary, 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." <http://gking.harvard.edu/files/gking/files/infoex.pdf?m=1360039060>, *International Organization* 57 (Summer): 617–642.
- Kovacs, Eduard. 2012a. "Iranian Official: We Did Not Launch Cyberattacks on American Banks." <http://news.softpedia.com/news/Iranian-Officials-We-Did-Not-Launch-Cyberattacks-on-American-Banks-294412.shtml>, *Softpedia Security News* (September 24).
- . 2012b. "Tick Tock: It's Lights Out for DNSChanger-Infected Computers on July 9." <http://news.softpedia.com/news/Tick-Tock-It-s-Lights-Out-for-DNSChanger-Infected-Computers-on-July-9-Video-279700.shtml>, *Softpedia Security News* (June 6).
- . 2013. "Hundereds of Sites Hacked in Conflict Between Malaysia and Philip-pines Hacktivists." <http://news.softpedia.com/news/Hundreds-of-Sites-Hacked-in-Conflict-Between-Malaysia-and-Philippines-Hacktivists-334047.shtml>, *Softpedia Security News* (March 4).
- Le, Quoc, and Tomas Mikolov. 2014. "Distrubuted Representations of Sentences and Documents." *Proceedings of the 31<sup>st</sup> International Conference on Machine Learning*.
- Lee, Sophie J., Howard Liu, and Michael D. Ward. 2016. "Lost in Space: Geolocation in Event Data." *arXiv:1611.04837* (November 14).
- Leetaru, Kalev, and Philip A. Schrodtt. 2013. "GDELT: Global Data on Events, Location and Tone, 1970-2012." <http://data.gdeltproject.org/documentation/ISA.2013.GDELT.pdf>. Annual Meeting of the International Studies Association.

- Lustick, Ian, Sean O’Brien, Steve Shellman, Timothy Siedlecki, and Michael Ward. 2015. “ICEWS Events of Interest Ground Truth Data Set.” Harvard Dataverse Network V3. <http://dx.doi.org/10.7910/DVN/28119>.
- Maaten, Laurens van der, and Geoffrey Hinton. 2009. “Visualizing High-Dimensional Data Using t-SNE.” *Journal of Machine Learning Research* 9 (Nov): 2579–2605.
- Mikolov, Tomas, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. “Efficient Estimation of Word Representations in Vector Space.” <http://arxiv.org/pdf/1301.3781.pdf>, *Proceedings of Workshop at ICLR*.
- Mikolov, Tomas, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. “Distributed Representations of Words and Phrases and their Compositionality.” <http://arxiv.org/abs/1310.4546>, *arXiv* (October).
- Nguyen, Kim Anh, Sabine Schulte im Walde, and Ngoc Thang Vu. 2016. “Integrating Distributional Lexical Contrast into Word Embeddings for Antonym-Synonym Distinction.” *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*: 454–459.
- Norris, Clayton, Philip Schrodtt, and John Beieler. 2017. “PETRARCH2: Another Event Coding Program.” *The Journal of Open Source Software* 2, no. 9 (January). doi:10.21105/joss.00133. <http://dx.doi.org/10.21105/joss.00133>.
- Open Event Data Alliance. 2015a. *PETRARCH Python Engine for Text Resolution and Related Coding Hierarchy*. Online. <http://www.github.com/openeventdata/petrarch>.
- . 2015b. “Phoenix Data Project.” [phoenixdata.org](http://phoenixdata.org).

- . 2015c. “Phoenix Pipeline.” <http://phoenix-pipeline.readthedocs.org/en/latest>.
- . 2018. “Universal Dependency PETRARCH.” <https://github.com/openeventdata/UniversalPetrarch>.
- Palmer, Glenn, Vito D’Orazio, Michael Kenwich, and Matthew Lane. 2015. “The MID4 dataset, 2002-2010: Procedures, coding rules and description.” *Conflict Management and Peace Science* 32 (2): 222–242.
- Pan, Sinno Jialin, and Qiang Yang. 2010. “A Survey on Transfer Learning.” *IEEE Transactions on Knowledge and Data Engineering* 22 (10): 1345–1359.
- Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, et al. 2011. “Scikit-learn: Machine Learning in Python.” *Journal of Machine Learning Research* 12:2825–2830.
- Pennington, Jeffrey, Richard Socher, and Christopher D. Manning. 2014. “GloVe: Global Vectors for Word Representation.” In *Empirical Methods in Natural Language Processing (EMNLP)*, 1532–1543.
- Radford, Benjamin J. 2017. “mmtsne: Multiple Maps t-SNE.” <https://cran.r-project.org/web/packages/mmtsne/index.html>, *CRAN: The Comprehensive R Archive Network*.
- Raleigh, Clionadh, Andrew Linke, Havard Hegre, and Joakim Karlsen. 2010. “Introducing ACLED - Armed Conflict Location and Event Data.” *Journal of Peace Research* 47 (5): 651–660.
- Raytheon BBN Technologies. 2015. “BBN ACCENT Event Coding Evaluation.”

- Rehurek, Radim, and Petr Sojka. 2010. "Software Framework for Topic Modelling with Large Corpora" [in English]. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*, 45–50. <http://is.muni.cz/publication/884893/en>. Valletta, Malta: ELRA, May.
- Rubenstein, Herbert, and John B. Goodenough. 1965. "Contextual Correlates of Synonymy." Edited by A. G. Oettinger. *Computational Linguistics* 8 (10).
- Santorini, Beatrice. 1990. "Part-of-Speech Tagging Guidelines for the Penn Treebank Project." [urlhttps://www.cis.upenn.edu/treebank/](https://www.cis.upenn.edu/treebank/).
- Schrodt, Philip A. 1998. "KEDS Kansas Event Data System Version 1.0." <http://eventdata.parusanalytics.com/>.
- . 2011. "TABARI: Textual Analysis by Augmented Replacement Instructions Version 0.7.6." <http://eventdata.parusanalytics.com/tabari.dir/tabari.manual.0.7.6.pdf>.
- Schrodt, Philip A., and David Van Brackle. 2013. "Automated Coding of Political Event Data." In *Handbook of Computational Approaches to Counterterrorism*, edited by V.S. Subrahmaniam. Springer Science + Business Media.
- Schrodt, Philip A, Deborah J Gerner, and Omur Yilmaz. 2009. "Conflict and mediation event observations (CAMEO): An event data framework for a post Cold War world." *International Conflict Mediation: New Approaches and Findings*.
- Swimmer, Morton. 2008. "Towards and Ontology of Malware Classes." <https://www.scribd.com/document/24058261/Towards-an-Ontology-of-Malware-Classes>.

- The Economist. 2012. "Hype and fear: America is leading the way in developing doctrines for cyber-warfare. Other countries may follow, but the value of offensive cyber capabilities is overrated." *The Economist* (December 8).
- The GDELT Project. 2016. "The Datasets of GDELT as of February 2016." <https://blog.gdeltproject.org/the-datasets-of-gdel-t-as-of-february-2016>. March 13.
- Valeriano, Brandon, and Ryan C. Maness. 2014. "The dynamics of Cyber Conflict Between Rival Antagonists, 2001-11." *Journal of Peace Research*.
- Volz, Dustin, and Jim Finkle. March 25, 2016. "U.S. indicts Iranians for hacking dozens of banks, New York dam." <http://www.reuters.com/article/us-usa-iran-cyber-idUSKCN0WQ1JF>, *Reuters*.
- Wang, Wei, Ryan Kennedy, David Lazer, and Naren Ramakrishnan. 2016. "Growing pains for global monitoring of societal events." *Science Magazine Digital* (September 30): 1502–1503.
- Ward, Michael D., Andreas Beger, Josh Cutler, Matthew Dickenson, Cassy Dorff, and Ben Radford. 2013. "Comparing GDELT and ICEWS Event Data." [http://mdwardlab.com/sites/default/files/GDELTICEWS\\_0.pdf](http://mdwardlab.com/sites/default/files/GDELTICEWS_0.pdf).