

# A Taxonomy of Ecosystem Management Actions

Timothy C. Haas  
Lubar College of Business  
University of Wisconsin at Milwaukee  
haas@uwm.edu  
October 8, 2023

## 1 Introduction

A taxonomy has been developed of actions taken by humans and by wildlife that affect an ecosystem such as a savanna or a mixed-use forest. This *Ecosystem Management Actions Taxonomy* (EMAT) possesses associated learning algorithms that allow it to be extended as new types of ecosystem-affecting actions are observed. A taxonomy is a necessary first step towards building a theory of how political-ecological systems function (Haas 2021). EMAT is a second-generation product built on an earlier attempt described in Haas and Ferreira (2018).

## 2 Automatic Acquisition of Political-Ecological Data

This document describes software both within and external to **id** for acquiring data off of the World Wide Web (web). Topics include the following.

1. Utilities for automatically discovering and downloading news articles (stories) from the web (so-called “web scraping”). These stories pertain to actions taken by individuals or groups that affect an ecosystem.
2. Utilities for automatically downloading ecological data such as remotely-sensed images, wildlife capture-recapture observations, and wildlife counts collected from surveys.
3. The EMAT.
4. EMAT action extraction procedure.
5. Within-sentence EMAT action extraction algorithm.

6. Procedure for assessing EMAT action extraction accuracy.
7. Learning new EMAT actions.

## 2.1 How to scrape news stories from the Web

Three methods have been developed for scraping political-ecological news articles (stories) from the World Wide Web. Several files containing code are mentioned herein. Download all batch, Powershell<sup>(TM)</sup>, and Outlook<sup>TM</sup> macro files to a single folder, e.g. `C:\pedatacq` (for *political-ecological data acquisition*). See Table 1 for a summary of these files.

File	Function
Manual aggregation	
<code>catstories.bat</code>	Calls <code>catstories.ps1</code> to add a <code>beginarticle 0</code> line to each story file, and then concatenate all such story files into one file.
<code>catstories.ps1</code>	As above.
Download Google Alerts	
<code>getalerts.bat</code>	Acquires Google Alerts stories.
Download Newslookup and Newsapi stories	
<code>getnews.bat</code>	Downloads news aggregator stories.
<code>getnews.ps1</code>	Does the scraping. Called by <code>getalerts.bat</code> and <code>getnews.bat</code>
Add parsed EMAT actions	
<code>phrases.bat</code>	

Table 1: Codes for web scraping. Batch files have a `.bat` extension, PowerShell programs, a `.ps1` extension, and Outlook macros, a `.txt` extension.

### 2.1.1 Method 1: Aggregating existing, separate story files

Use a file naming scheme that allows the collection of files to be referred to with *wildcard* notation. For example, files named as *story1.txt*, *story2.txt*, ... may be collectively referred to as *story\*.txt*. Running the batch file, `catstories.bat` will result in the line `beginarticle 0` being inserted as line 1 in each file, followed by all of these files being concatenated into one file. Edit this batch file to specify the collection of files to be aggregated and the aggregation file as follows.

```
catstories wildcard_filenames append_file_name
```

### 2.1.2 Method 2: Reading Google Alerts

1. Create a Google<sup>TM</sup> account and then create news alerts for a set of desired keyword phrases. Have these alerts sent to a mailbox.
2. Read each alert email. If a story seems relevant, open the story's link and write it to a file as an HTML-only file type ("webpage, HTML only" in Internet Explorer<sup>TM</sup>). Use a filename of *sn.htm* where *n* is a number, e.g. *s1.htm s2.htm* etc.
3. Prepare the **id** language file by listing the files to be read by entering the filename prefix, postfix, the number of files to be read, the starting DAN, and the file containing group names.
4. Add the action entries written to the output file to the actions data base, e.g. **eastafacts.dat**.
5. The collection of news stories with all HTML markup tags removed will be contained in a file named *dansntom.dat* where *n* is the starting DAN, and *m* is the ending dan in the file, respectively. In this file, each new story has a header line of the form

STORY: ——— start DAN= *n* end DAN= *m* ———

Stories for which the relation's parsing algorithm failed are written to the file **parsefailed.dat**.

To acquire each story pointed to within a Google Alert email, you need to do the following:

1. Manage alerts.
2. Download all alert emails in a folder as either a single file, or a collection of files.
3. Open and download each original hyperlink in either a file of concatenated Google Alert emails or a collection of separate email files.

### How to manage Google Alerts

To add or delete a Google News Alert, log into your Google account by accessing the link [www.google.com/accounts](http://www.google.com/accounts). Your username is your email address. Then, click **Dashboard** (click **Accounts** and then click **Alerts**).

## Put Google Alerts in a folder

In Outlook Web Access (OWA), click **Settings** and then **Options**. Or, click **Settings**, **Show all Settings**, and then **Options**. Worst case, you may have to enter **options** as a search word to get to that item. Once there, click **Inbox Rules**.

The manual method for downloading an OWA folder is as follows.

1. Start a local Outlook session and login to the connected OWA. Let the local Outlook application settle its updating issues before doing the above. It may be that the local Outlook application decides that many of the Google Alerts emails are junk. It will take awhile but it will eventually move these to the Junk folder on both the local and OWA applications. Don't be alarmed as no emails will be deleted. Once settled, follow the above steps for downloading Alerts emails from the Junk folder after you have performed a search of the Junk folder for "Google Alerts" emails.
2. If not present as a dotted line icon in the very top row (left) of the local Outlook application's icons, add the **Select All** command to its Toolbar. Do this by selecting **All Commands** in the **Choose Commands** list, scrolling to the **Select All** command, and then clicking **Add**.
3. Next, get into the desired folder, e.g. `googleeestaf` and then click the **Select All** icon.
4. After that, click **File** and then **Save As**. Take the default (**Text Only**).

Do not click **more files on Exchange** because for some reason, the URL content is not written to the Save As file.

5. Rename this file to have an extension of `.dat`. Call this the *alerts file*.

On day 5 of each month, the `getalerts.bat` batch file runs from Task Scheduler. If a user does not respond to the batch file's prompts, the batch job will terminate. In this case, the user should manually run this batch job by typing `C:/pedatacq/getalerts` at a command line prompt. User interaction is required. **Only after starting this batch job**, the user needs to start a local version of Outlook. Then, the user needs to do the following:

1. Click **Developer**, **Macros**, and then `Project1.SaveAllEmails_ProcessAllSubFolders`.
2. Choose the folder to be downloaded, e.g. `googleeestaf`.

3. After the macro finishes, verify that this folder is now empty on both the desktop version of Outlook and on OWA.
4. Hit any key to continue the `getalerts` batch job.

This batch file continues by scraping the stories off the web pointed to in the downloaded emails.

**Concatenating and cleaning Alerts files** The batch file `getalerts.bat` performs the following steps.

1. Concatenate all Google Alerts emails into one alerts file.
2. Clean this file by first, removing `^@` control characters via the device of converting the file from UTF-16 to ANSI. And secondly, replacing the beginning and ending string added by Google and Outlook in every news story URL with a space so that `getnews.ps1` can successfully read and then connect to these URLs. These replacements need to be performed in the following order.

1. Replace `%3A` with `:`
2. Replace `%2F` with `/`
3. Replace `%3D` with a space
4. Replace `%26` with a space
5. Replace

`<https://nam02.safelinks.protection.outlook.com/?url=`

with a space.

6. Replace `https://www.google.com/` with a space.
3. Download each story pointed to by a link in the Google Alerts by executing the following command.

```
powershell c:/pedatacq/getnews.ps1 googlealerts alerts_filename  
                                         append_filename
```

The second argument to this command is the alerts file, and the third is the file to append the downloaded stories to.

### 2.1.3 Method 3: Use of commercial news aggregators

This method uses a commercial news aggregator to find stories.

The PowerShell program `getnews.ps1` is reused from above to scrape stories from the commercial news aggregator, `newsapi.org`. You'll need to first setup a free account with `newsapi.org`. The program `getnews.ps1` is run within the batch file `getnews.bat`.

To run this every Friday at 10:28am, use Task Scheduler, found at **Windows Administrative Tools > Task Scheduler**. To edit a task, click **Task Library** and then click on the task. Then, click **Properties** from the menu on the right of the screen. Then, click the characteristic of the task you wish to edit and then click **Edit**. For this task's Action, specify **Run a Program** and then enter the full path to the program `getnews.bat`, e.g. `c:/pedatacq/getnews.bat`. Set the **General** options of the task so that **runs only when the user is logged on** is checked.

## 2.2 How to download ecological data

Section to be developed.

## 3 The EMAT

Each EMAT entry can be decomposed into three sentence components: *m-word verb*, a direct object phrase, and/or a prepositional phrase. Letting *m* be a positive integer, an *m-word verb* subsumes single-word verbs (either *regular* or *irregular*), and *multi-word verbs* (these use more than one word to convey their meaning, e.g. "picked up"). A multi-word verb is also referred to as a *phrasal verb* (Spears 2006).

These decompositions are realized in a separate file wherein each EMAT entry has been manually parsed into three *equivalence sets*: a set of semantically equivalent *m-word verbs*, a set of semantically equivalent direct object phrases, and a set of semantically equivalent prepositional phrases, respectively. Initial entries in an action's equivalence sets are phrases from stories that report unambiguous instances of the action. For *m-word verb* equivalence sets in particular, if an EMAT action's *m-word verb* equivalence set contains a 1-word regular verb, then all conjugated forms of that verb are also included in the set. Such regular verbs are added to `parsedemataacts.dat` in their past-tense form. Doing so supports the following automatic verb conjugation algorithm.

When a 1-word verb ending in "ed" is read into an equivalence set, all conjugated forms of that verb are added to the *m-word verb* equivalence set for that

action. For example, when the 1-word verb, *poached* is read, the 1-word verbs *poach*, *poaching*, and *poaches* are immediately added to the equivalence set.

Phrasal verbs can be partially conjugated by their 1-word verb substitutions, if any (Bryson 2022). For example, “The campers were thirsty after they used up the last of the water.” can be rewritten as “The campers were thirsty after they exhausted the last of the water.”

A taxonomy such as the EMAT is an ontology that represents only hierarchical relationships among its members. The American Society for Indexing (American Society for Indexing (2018)) describes a taxonomy as

“a tree-hierarchical controlled vocabulary lacking more complex relationships found in thesauri or ontologies”

but point out that

“An ontology is a kind of taxonomy with structure and specific types of relationships between terms. In an ontology the types of relationships are greater in number (than in a taxonomy) and more specific in their function. Information that in a taxonomy is conveyed through indexing, is embedded into the ontology itself.”

A foundational paper on the *Semantic Web* (see Yu (2015)) describes a taxonomy as a *simple ontology* (McGuinness 2005) (see also van Rees (2003)). Taxonomies are also known as *hierarchical ontologies*, see Khan (2014). A taxonomy then, can be viewed as a simple form of an ontology.

On the ecological side, this article’s database contains entities that describe the status of an ecosystem. These are referred to as *ecological actions*. These actions are of various types including species abundance, habitat metrics such as vegetation index, wildlife disease outbreak events, events of wildlife-caused damage to crops, and events of wildlife attacks on humans. The database holds observations of these actions that have been gleaned from a variety of sources including stories, pre-analyzed remotely sensed images, and published surveys of wildlife abundance. A taxonomy can be used to map a potentially large number of possible political-ecological actions to a finite number of actions that cover most things that humans can do to an ecosystem along with the ways an ecosystem can respond to those actions. Having a finite set of possible actions allows models of the interactions between humans and ecosystems to be specified in terms of a set of archetypal political actions taken by ecosystem-affecting actors, and archetypal ecological actions taken by the ecosystem in

response to those political actions. This finite set of possible political-ecological actions then, does not grow with the size of a political-ecological actions data set (sample size) – resulting in models that are based on such political-ecological action data sets that are parsimonious and hence, mappable to theoretical constructs in sociology, political science, and ecology.

One such taxonomy, the EMAT, is an extension of a political actions taxonomy developed by Leng (1993). Table 2 contains a few of the more frequently encountered EMAT actions. Currently, the EMAT consists of 119 militaristic actions, 191 diplomatic actions,

Category	Subcategory	Action	ID	Archetypal Actor → Target
Political	Military	Arrest some poaching suspects	MM0015	B → H
	Diplomatic	Petition to stop wildlife-caused crop destruction	D12719X3	F → A
	Economic	Complete electric wildlife-control fence	E23719X3	E → H
	Ecosystem directed	Translocate animals	C0008	B → K
	Ecosystem directed	Poach some elephants	CED15	F → K
Ecosystem		Elephants trample crops	Z006	K → F

Table 2: Frequently encountered EMAT actions. Archetypal actors and targets are denoted as: A = president, B = EPA, E = EPA or NGO, F = rural resident, H = rural resident or pastoralist, and K = ecosystem. Archetypal actors and targets are based on encounters with actual stories. The archetypal groups are used only when the EMAT action extraction algorithm fails to find mention of both actor and target in a story.

198 economic actions, 92 ecosystem-directed anthropogenic actions, and 37 ecological actions. Each action is associated with a set of archetypal actors. For example, the action **Complete electrified wildlife control fence** can be executed by either an environmental protection agency (EPA) or a non-governmental organization (NGO). Here, “EPA” is a generic moniker for any governmental agency charged with protecting wildlife and/or the environment. Each EMAT action is implicitly associated with a particular scale of influence. For example, the scale of the EMAT action, **Poach for food** is regional, whereas the scale of the EMAT action **Strengthen wildlife protection laws** is national.

### 3.1 Taxonomy validity

A taxonomy should contain entries that are orthogonal to each other. Its collection of entries should represent most of the possible actions in the phenomenon it has been designed



to detect in the real world (Unterkalmsteiner and Adbeen 2022). One way to measure the distance between EMAT actions is to compute their pairwise *Levenshtein distance*.

## 4 Acquiring EMAT action observations

Links between EMAT actions and actions reported in stories are discovered by running a parsing algorithm that is programmed in **id**.

**id** can extract actions at scale by analyzing stories in parallel. This is accomplished by taking advantage of the *embarrassingly parallel search* (see Malapert et al. (2016)) characteristic of the action extraction procedure: The extraction of actions from one story is independent of the extraction of actions from some other story. Therefore, a cluster computer having  $m$  compute nodes, each with at least four processors can deliver a near-linear speed up in the processing time of  $n > m$  stories.

### 4.1 Action acquisition procedure

EMAT action observations are acquired from stories via the following procedure.

1. Cluster the  $m$  EMAT actions such that the min-max similarity score between one action’s  $m$ -word verb equivalence set and another’s is greater than 0.98 when both of these actions belong to the same cluster. Label these clusters *action cluster  $i$* ,  $i = 1, \dots, q$  where  $q$  is the number of clusters. Note that  $q \leq m$ .
2. Acquire a file of raw HTML stories. The automatic system used here that scrapes stories from the web is described in Haas (2018).

Run the **id** relation, **parse\_stories** to extract sentences,  $m$ -word verbs, direct object phrases, prepositional phrases, and EMAT action observations from these stories. For example, the following **id** input file extracts actions from the story file, **ef9-13.txt**.

```
report prepare_data
  parse_stories(2
    c:/polbio/stories/ef9-13.txt
    false false 1 efgroups.dat efgroups.dat
    ef9-13sdb.dat ef9-13acts.dat)
```

This relation executes the following steps.

- (a) Scan each story for the story’s source.

- (b) Remove a pre-selected set of HTML tags from the story to produce a *tag-filtered story*. This step is performed after scanning for the story’s source because experience has shown that often, a story’s source is contained inside unforeseen HTML tags that are not part of the story’s main text.
- (c) Form a text fragment of the story that consists of textual content sentences only. A sentence contains textual content if (a) it contains at least three *common words* defined by the list {*the, a, of, is, by, to, be, from, and, have, in, that, on, with, as, at, inside*}; and (b) less than 80% of its words are *irrelevant* as defined by the list {*content, copyright, stylesheet, subscribe, subscription, login, header, sidebar, wrapper, label, navigation, class, column, http:, republish, div*}.

This author has read the raw HTML of several hundred online news articles posted by news organizations located in the United States, many African countries, many European countries, and many Asian countries. This experience has shown that a story’s text and attendant identifiers are often scattered across several HTML tags rather than being located within predictable tag sets. The systems used to generate the raw HTML apparently follow no standard and are of highly variable quality.

Several off-the-shelf packages for parsing HTML have achieved mixed success in finding all the needed elements of a story when applied to such a varied range of HTML writing style. Because of this deficiency and because of a desire to create a self-contained, single software system for acquiring and analyzing political-ecological data, this author has programmed the above story-extraction algorithm into the **id** software system. Trial and error has guided the choice of words for the common words list, and the irrelevant words list. Trial and error has also led to the setting of the “less than 80% irrelevant words” rule.

- (d) Apply a second search for narrative text by having the Jsoup parser (ref??) extract sentences from the above text fragment.
- (e) Search each tag-cleansed story for its date, groups, regions, and actions. These four searches are performed simultaneously by running each search in its own, independent *thread*. A speed-up will accrue when the code is run on a computer or node that has at least four processors. The *EMAT action extraction algorithm* (see the next Section) is used to search for actions.

As improvements to the EMAT action extraction algorithm are made and/or associated lexicon files are updated, political-ecological databases based on actions extracted from

these stories will need to be-created by re-running the relation on the underlying raw HTML stories.

## 4.2 EMAT action extraction algorithm

This algorithm searches a sentence for  $m$ -word verbs that *partially match*  $m$ -word verbs that are members of an EMAT action’s  $m$ -word verb equivalence set. For example, say that some hypothetical story contains the sentence

Five poachers were arrested on June 10, 2019 and sentenced to prison on August 8, 2019.

This sentence contains two 1-word verbs: *arrested*, and *sentenced*. Similar searches are executed to find direct object phrases, and prepositional phrases that partially match members of corresponding equivalence sets. See OpenOregon (2023) for a review of prepositional phrases. Parsing is accomplished with a modified version of the shallow parsing algorithm of Daelemans et al. (1999).

### 4.2.1 Algorithm

1. Using the *phrase similarity* sub-algorithm described below, search a sentence for an  $m$ -word verb that best matches entries in  $m$ -word verb equivalence sets. Declare a match if a pair’s similarity ( $SIM$ ) is greater than 0.95. Let this best-matching EMAT action be a member of action cluster  $j$ . Phrases are allowed to appear in any order within a sentence.
2. If no matches are found, return null.
3. Initialize the variables  $SIMD$  and  $SIMP$  to 0.0.
4. Search the sentence for a direct-object phrase that matches an entry in direct-object phrase equivalence sets of EMAT actions in action cluster  $j$ . Declare a match if  $SIM > 0.95$ . Store the matched EMAT action with the highest similarity score in  $actionD$  and store its similarity value in  $SIMD$ .
5. Search the sentence for a prepositional phrase that matches an equivalence set prepositional phrase of one of the actions in action cluster  $j$ . Declare a match if  $SIM > 0.95$ . Store the matched EMAT action with the highest similarity score in  $actionP$  and store its similarity value in  $SIMP$ .

**\*\*Tip:** If there are critical words that need to be detected in an action, keep the prepositional phrase short with the needed word. That way, the similarity measure will be large only if that word is present.

6. If *actionD* and *actionP* are both null, return null.
7. If *SIMP* > 0.8 return *actionP*.
8. If *SIMD* > *SIMP*, return EMAT *actionD*. Otherwise, return null.

### Sub-algorithm

In what follows, an *n-gram* is a subsequence of *n* words in a natural language phrase. One definition of the degree of similarity between two phrases is the *Phrasal Overlap Measure* of Ponzetto and Strube (2007). A modified version of this measure follows.

Notation:

1. Without loss of generality, let the number of words in phrase 1 be  $N = |ph_1| \leq |ph_2|$ .
2. Let *s* be the number of times *n*-gram pairs are formed by starting at the same location in each phrase.
3. Let  $m_n$  be the number of *n*-grams that are common to the two phrases. A pair of *n*-grams are declared to be common if 1.0 minus the *Levenshtein distance* (Levenshtein (1996), Yujian and Bo (2007)) between the two is greater than 0.99.

The modified measure is:

$$SIM(ph_1, ph_2) \equiv (N/|ph_2|) \tanh \left[ \frac{1}{s} \sum_{n=1}^N m_n n^2 \right]. \quad (1)$$

If  $|ph_1| = |ph_2| = 1$ ,  $SIM(ph_1, ph_2)$  is 1.0 minus the Levenshtein distance between the two.

This measure of phrase similarity is interpretable because it lies in the unit interval. This algorithm is a new version of one originally developed in Haas (2018).

## 4.3 Assessing extraction accuracy

Haas (2018) discusses algorithms designed to extract taxonomic actions from media and concludes that such an algorithm

should be evaluated on its accuracy and speed. The algorithm’s accuracy can be assessed by comparing the actions extracted from a random sample of stories to those extracted by a human reading the same set of stories. Using the set of human-extracted actions as the benchmark, the algorithm can make two types of errors: failing to extract an action in a story; and extracting an action that does not exist in the story, referred to here as a *spurious action*.

Let  $n_{true}$  be the number of human-extracted actions from a story file. Let  $r_{correct}$  be the fraction of  $n_{true}$  that were extracted by the algorithm. Let  $r_{spurious}$  be the ratio of the number of spurious actions to  $n_{true}$ .

When `parse_stories` executes, an *extraction assessment* file is written that contains ten story texts along with those actions extracted by the algorithm. These texts have been uniformly sampled from the raw HTML story file. The values of  $r_{correct}$  and  $r_{spurious}$  can be computed by manually reading this extraction assessment file and noting the number of true actions; the number of extracted actions that match true actions; and the number of extracted actions that are spurious.

?? program this.

### 4.3.1 Spot check

1. Run the *parse\_stories* accuracy check on each raw stories file.
2. Concatenate all `*check.dat` files into a single file.
3. Bring this file into the Chromebook viewer.
4. Find the total number of lines in this file,  $m$  and then read a story every  $m/10$  lines.
5. Record  $n_{true}$ ,  $n_{correct}$ , and  $n_{spurious}$  for each of these stories.
6. Along the way, record frequent spurious actions and actions that are not detected in order to improve the algorithm by editing `parsedematactions.dat`.
7. From these values, compute an estimate of the algorithm’s accuracy.

## 5 Learning new EMAT actions

EMAT actions and their equivalence sets do not define a static taxonomy but rather a dynamic one as both the language evolves and new interactions between humans and

ecosystems emerge. This dynamic characteristic of the EMAT is operationalized with a learning algorithm that can identify either a new equivalence set member of an existing EMAT action or, more fundamentally, an entirely new EMAT action. This algorithm is new and is derived in-part, from a general outline for one given in Haas (2018).

## 5.1 Semi-automatic learning algorithm

1. Run the DOS batch file, `C:\polbio\stories\phrases.bat` on short strings (1 to 2 words long) that have something to do with the EMAT action. View the output file for common usage in the raw HTML files of these strings. These instances will suggest *m*-word verbs, objects, and prepositional phrases to add to the file `c:\polbio\parsedemataacts.dat`.
2. Detect those sentences in a set of stories that have a maximum overall similarity score greater than 1.6 but less than 1.9, the value needed to declare an observed EMAT action. Create a list of these {**sentence**, **EMAT action**} pairs. Recall that an overall similarity score expresses the similarity of a sentence to a particular EMAT action. Hence, the maximum overall similarity score is always associated with a particular EMAT action.
3. Examine each sentence in this list to determine if it is clearly describing an occurrence of any existing EMAT action. If not, go to Step 3. Otherwise, add the *m*-word verb, direct object phrase, and prepositional phrase from this sentence to the equivalence sets of the most appropriate, existing EMAT action. Stop.
4. Examine the sentence for an ecosystem-relevant militaristic, diplomatic, economic, ecosystem-directed, or ecological action. If a new EMAT action for this action is judged to be needed, use the sentence's *m*-word verb, direct object phrase, and possibly the sentence's prepositional phrase as the initial members of this new action's three equivalence sets, respectively. Stop.

## 5.2 Example of discovering a new equivalence set member

The following sentence gives a high overall similarity score for the EMAT action **Sell a few rhino horns**.

“In 2014, Kenya enacted tough new laws that make ivory poaching and trafficking punishable by fines of \$200,000 or even life in prison compared to the maximum fines of about \$400 that were handed out previously.”

Clearly, this sentence describes an observation on the EMAT action: *tighten wildlife agreement or laws*. Hence, the 1-word verb *enacted* should be added to this action’s *m*-word verb equivalence set, and the phrase *tough new laws that make ivory poaching and trafficking punishable* should be added to the action’s direct object phrase equivalence set.

### 5.3 Example of discovering a new EMAT action

The following sentence from the file `ef-script15-16.txt` produces a high overall similarity score on the EMAT action `Sell a few rhino horns` but is clearly not describing that or any other existing EMAT action.

“TenBoma is a communications based initiative that uses modern technology and sophisticated data analysis to allow law enforcement agencies to predict poaching plots in advance and thwart the incidents.”

Instead, this sentence appears to be describing a new ecosystem management action that is about the introduction of a new technology to combat wildlife trafficking. Hence the action and associated equivalence set members as shown in Table 3 should be added to the EMAT.

Database entity	Value of its <i>phrase</i> attribute
Action	Develop new technology to combat wildlife trafficking
<i>m</i> -word verb	<i>uses</i>
direct object phrase	<i>modern technology and sophisticated data analysis</i>
prepositional phrase	<i>to predict poaching plots</i>

Table 3: A new EMAT action along with its initial equivalence set members.

## 6 Stories Repository

1. A file with a `.txt` extension is a story file that has been scraped off the web with minimal attempts at removing HTML fluff. These are located in the directory `c:/polbio/stories`.
2. A file with a `.dat` extension is a file that contains either political-ecological actions database entities (one set of entities per story) or contains an actions history. Both of these file types are generated when the `id` relation, `parse_stories()` is run on story files.

### 6.1 Schedule and procedure

??How to add more search terms to the `newsapi` command in contained in `getnews.bat`?

1. Acquire Google Alerts stories and collect them into a story file.
2. Run the `parse_stories()` command in `id` on story files contained in the directory `c:/polbio/stories`.

### 6.2 Story locations

See Table 4 for a list of the repository’s story files. The batch file, `getnews.bat` generates a file containing raw HTML of online news stories about rhino poaching. These files are named `newsapi202ymmdd.txt` where “2027mmdd” is the year-month-date of the week’s Friday on which the program was run. At the beginning of each year, last year’s files are concatenated with the following commands:

```
cat newsapi202x*.txt > dum.txt
```

followed by

```
move dum.txt newsapi202x.txt
```

where “202x” is last year, e.g. 2023.



Size	Story file	Story DB file	Actions file
East Africa			
38,675,592	ef9-13.txt	ef9-13sdb.dat	ef9-13acts.dat
291,897,345	ef13-14.txt	ef13-14sdb.dat	ef13-14acts.dat
190,719,056	ef14-15.txt	ef14-15sdb.dat	ef14-15acts.dat
190,276,554	ef15-16.txt	ef15-16sdb.dat	ef15-16acts.dat
120,033,188	ef16-19.txt	ef16-19sdb.dat	ef16-19acts.dat
123,923,566	ef19.txt	ef19sdb.dat	ef19acts.dat
461,581,001	ef19-21.txt	ef19-21sdb.dat	ef19-21acts.dat
763,133,494	alerts21.txt	knp21asdb.dat	knp21aacts.dat
2,661,556,915	alerts22.txt	knp22asdb.dat	knp22aacts.dat
Kruger National Park			
94,489,580	knp13-20.txt	knp13-20sdb.dat	knp13-20acts.dat
65,718,985	newsapi21.txt	knp21bsdb.dat	knp21bacts.dat
189,123,060	newsapi22.txt	knp22bsdb.dat	knp22bacts.dat

Table 4: Story files in `c:/polbio/stories`. The file name pattern “...sdb.dat” is a story database file, and “...acts.dat” is an actions history file.

### 6.2.1 Accuracy assessment

Table 5 contains extraction accuracy assessments for each file in the story repository.

Story file	$n_{true}$	$r_{correct}$	$r_{spurious}$
East Africa			
ef9-13.txt	19	.74	.37
ef13-14.txt			
ef14-15.txt			
ef15-16.txt			
ef16-19.txt			
ef19.txt			
ef19-21.txt			
alerts21.txt			
alerts22.txt			
Kruger National Park			
knp13-20.txt			
newsapi21.txt			
newsapi22.txt			

Table 5: Estimates of the accuracy of the **id** action extraction algorithm. An estimate is computed from 10 stories uniformly sampled across a file.

# References

- American Society for Indexing (2018). *Taxonomies & Controlled Vocabularies Special Interest Group*. Available online: [www.taxonomies-sig.org/about.htm](http://www.taxonomies-sig.org/about.htm) (accessed on 2 July 2018).
- Bryson, S. (2022), *List of 47 Phrasal Verbs and Their One-Word Substitutions*, <https://www.scribbr.com/academic-writing/phrasal-verb-alternatives/>
- Daelemans, W.; Buchholz, S.; Veenstra, J. (1999). Memory-Based Shallow Parsing. In *Proceedings of the EACL'99 Workshop on Computational Natural Language Learning (CoNLL-99)*, Bergen, Norway, 53-60.
- Haas, T. C. (2001), "A Web-Based System for Public-Private Sector Collaborative Ecosystem Management," *Stochastic Environmental Research and Risk Assessment*, 15(2): 101-131.
- (2002), "New Systems for Modeling, Estimating, and Predicting a Multivariate Spatio-Temporal Process," *Environmetrics*, 13(4): 311-332.
- (2004) "Ecosystem Management via Interacting Models of Political and Ecological Processes," *Animal Biodiversity and Conservation*, 27(1): 231-245. [www.bcn.es/museuciencias](http://www.bcn.es/museuciencias)
- (2011), *Improving Natural Resource Management: Ecological and Political Models*, a "Statistics in Practice" volume, cross-listed in the Environmental Management, Policy and Planning series, and the Environmental Economics and Politics series, Wiley-Blackwell, Oxford, U.K. ISBN: 978-0-470-66113-0. [www.wiley.com/WileyCDA/Section/id-350698.html](http://www.wiley.com/WileyCDA/Section/id-350698.html)
- (2018). Automatic acquisition and sustainable use of political-ecological data. *Data Science*, 17, p.17, DOI: 10.5334/dsj-2018-017 (Accessed August 1, 2018).
- (2019). *Rhino Ecosystem Management Tool*,. Available online: [https://sites.uwm.edu/haas/home/rhino\\_emt](https://sites.uwm.edu/haas/home/rhino_emt) (Accessed May 11, 2021).
- (2021), "The First Political-Ecological Database and its Use in Episode Analysis," *Frontiers in Conservation Science*, section: *Planning and Decision-Making in Human-Wildlife Conflict and Coexistence*, 2:707088. doi: 10.3389/fcosc.2021.707088 <https://www.frontiersin.org/article/10.3389/fcosc.2021.707088>
- , Mowrer, H. T., and Shepperd, W. D. (1994), "Modeling Aspen Stand Growth with a Temporal Bayes Network," *Artificial Intelligence Applications*, 8(1), 15-28.
- and Ferreira, S. M. (2018), "Finding Politically Feasible Conservation Strategies: The

- Case of Wildlife Trafficking,” *Ecological Applications*, 28(2): 473-494, DOI: 10.1002/eap.1662.
- Khan, S.; Safyan, M. (2014). “Semantic matching in hierarchical ontologies,” *Journal of King Saud University - Computer and Information Science*, 26(3), 247-257.
- Leng, R. J. (1999). *Behavioral Correlates of War, 1816-1979* (Computer File), 3rd Release, Middlebury College, Middlebury, VT, 1993, Study Number 8606 from the Inter-University Consortium for Political and Social Research (ICPSR), Ann Arbor, Michigan, USA. Available online: [www.icpsr.umich.edu/icpsrweb/ICPSR/studies/8606](http://www.icpsr.umich.edu/icpsrweb/ICPSR/studies/8606) (accessed on 2 July 2018).
- Levenshtein, A. (1966). “Binary codes capable of correcting deletions, insertions and reversals,” *Soviet Physics Doklady*, 10(8), 707-710.
- Malapert, A., Régim, J. C., and Rezgui, M. (2016). Embarrassingly parallel search in constraint programming. *Journal of Artificial Intelligence Research*, 57, 421-464.
- McGuinness, D. L. (2005). “Ontologies come of age.” In *Spinning the Semantic Web: Bringing the World Wide Web to its Full Potential*; Fensel, D., Hendler, J., Lieberman, H., Wahlster, W., Eds.; The MIT Press: Boston, MA, pp. 171-194; 978-0262562126.
- OpenOregon (2023) *10.3 Grammar: Prepositional Phrases*, <https://openoregon.pressbooks.pub/synthesis/chapter/10-3-grammar-using-prepositional-phrases/>
- Ponzetto, S. P.; Strube, M. (2007). “Knowledge derived from Wikipedia for computing semantic relatedness,” *Journal of Artificial Intelligence Research*, 30, 181-212.
- Spears, R. A. (2006), *McGraw-Hill’s Dictionary of American Idioms and Phrasal Verbs*, First Edition, McGraw-Hill, New York, 1104 pages, ISBN: 0071469346.
- Unterkalmsteiner, M. and Adbeen, W. (2022), “A Compendium and Evaluation of Taxonomy Quality Attributes,” *Expert Systems*, 40:e13098, DOI: 10.1111/exsy.13098.
- van Rees, R. (2003). Clarity in the Usage of the Terms Ontology, Taxonomy and Classification. In *CIB W78’s 20th International Conference on Construction IT, Construction IT Bridging the Distance*, Amor, R.; Ed.; Waiheke Island, New Zealand, 23-25 April. Available online: <http://itc.scix.net/cgi-bin/works/Show?w78-2003-432> (accessed on 2 July 2018).
- Yu, L. (2015). *A Developer’s Guide to the Semantic Web*, 2<sup>nd</sup> ed.; Springer: Heidelberg, Germany, 978-3662437957.

Yujian, L., Bo, L. (2007). “A normalized Levenshtein distance metric.” *IEEE Transactions on Pattern Analysis and Machine Intelligence*, June, 29(6), 1091-1095.