User's Guide to Utilities for Automatic Acquisition of Political-Ecological Data

Timothy C. Haas
Lubar School of Business
University of Wisconsin at Milwaukee
haas@uwm.edu
July 2, 2019

1 Overview

Section 2 of this User's Guide documents several utilities for automatically discovering and downloading news stories from the World Wide Web (so-called "web scraping"). These stories pertain to actions taken by individuals or groups that affect an ecosystem. Section 3 documents utilities for automatically downloading ecological data such as remotely-sensed images, wildlife capture-recapture observations, and wildlife counts collected from surveys.

Download all batch, VBScript$^{(TM)}$, and JAVA$^{(TM)}$ files to a single folder. For example, `C:\pedatacq`.

2 Scraping News Stories from the Web

Three methods have been developed for scraping political-ecological news articles from the World Wide Web.

2.1 Method 1: Aggregating existing, separate story files

It may be necessary to first create a list of filenames in this collection of HTML story files. One way to do this is with the command `dir /b *.txt > filelist.txt`. Remember to remove from `filelist.txt` any filename that is itself a list of filenames.

Then, the main task may be executed: Adding an article number to each story and concatenating all of these stories into one file. Two ways to do this have been developed.

2.1.1 Concatenation procedure 1

1. If needed, add a "beginarticle" line to each story file. One way to do this is with the sed editor command:
   `sed -i -e 1ibeginarticle -s stories*.txt` for a collection of story files all starting with the word "stories." The batch file, `addline.bat` contains this command.

2. Concatenate these files. For example, `cat s0*.txt > all.txt`,

3. It will be necessary to add a dummy article number to each story in `all.txt` with the vim command `g/beginarticle/s//beginarticle 0/g`.

2.1.2 Concatenation procedure 2

Run the command `catstories.vbs`. This program (a) adds an article number to each story, (b) removes some of the HTML fluff, and (c) concatenates separate story files into one file by opening each file listed in the text file *file-list-name* and appending it to the text file *append-file-name*. The arguments to `catstories.vbs` are:
file-list-name, append-file-name.

2.2 Method 2: Use of a commercial new aggregator

This method uses a commercial news aggregator to find stories. The program `getnews.vbs`: scrapes these stories from `newslookup.com` and appends them to a file. You'll need to first setup a free account with `newslookup.com`. The program `getnews.vbs` is run within the DOS macro program (batch file), `kruger.bat`. This batch file is as follows.

```
rem To run this every week at 10:28am, use Task Scheduler, found at
rem Accessories > System Tools.  For the Action, specify "Run a Program"
rem and then enter just "kruger.bat" (no path).  Then, set the
rem optional "Start In" value to the directory this batch program is in,
rem for example, "c:\polbio\sanparks\"
rem
copy /y kruger5417.txt krugerback.txt
tail -2 kruger5417.txt > lastline.dat
```

```
rem
copy /y c:\polbio\stories\getnews.vbs
cscript getnews.vbs "kruger5417.txt"
rem del lastline.dat
```

2.3 Method 3: Reading Google Alerts

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, e.g. a collection of .eml files.

2.3.1 How to manage alerts

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

2.3.2 Put Google Alerts in a folder

In Outlook Web App (OWA), click Settings and then Options. You may have to enter "options" as a search word to get to that item. Then, click "Inbox Rules."

2.3.3 How to download an Outlook OWA folder

1. Start a local Outlook 2010 session and login to the connected OWA.
2. If not present as a dotted line icon in the very top row (left) of the local Outlook icons, add the `Select All` command to the Toolbar of Outlook. 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, and then click the `Select All` icon.
4. After that, click `File` and then `Save As`. Take the default (`Text Only`).

NOTE!!: Let the local Outlook settle its updating issues before doing the above. It may be that the local Outlook 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 platforms. Don't be alarmed: 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.

2.3.4 How to download the raw HTML of all Google Alerts

The program `getalerts.vbs` loads each page pointed to by a Google Alert email and downloads its raw HTML to an append file. This step consists of following the links inside the Google Alerts emails to the originating news stories. Execute `getalerts.vbs` by running the batch file `getstories.bat` after editing it so that the first argument is the append filename, and the second argument is the text file of all the Google Alert emails concatenated together.

2.4 How to Use id to process stories

Have **id** execute the `parse_stories` relation on a raw HTML story file. For example:

```
report prepare_data
   parse_stories(c:/polbio/stories/efstories-script15-16.txt false false 1
      efgroups.dat efregions.dat ef15-16prsd.dat ef15-16acts.dat)
```

3 How to Download Ecological Data

??

4 The EMAT

An EMAT action's equivalence sets are formed in two ways. First, stories that report unambiguous instances of the action are read in detail. Based on this reading, the m-word verb and phrases used in those stories are used for the initial entries in the action's equivalence sets. Secondly, for m-word verb equivalence sets, if an EMAT action's m-word verb equivalence set contains a 1-word verb that is regular, then all conjugated forms of

that verb are also included in the set. More sophisticated methods of including variants of a 1-word verb such as the practice of *stemming* (Porter (1980)) could be used to further enlarge the EMAT's m-word verb equivalence sets.

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). Consistent with this view then, that a taxonomy is 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 sample size – resulting in models based on this set of political-ecological actions that can be made parsimonious and hence, potentially mappable to theoretical constructs developed in the social sciences and ecology.

One such taxonomy, the EMAT is an extension of a political actions taxonomy developed by Leng (1993). Table 1 contains a few of the more frequently encountered EMAT actions. The EMAT consists of 119 militaristic actions, 191 diplomatic actions,

Category	**Subcategory**	**Action**	**ID**	**Archetypal Actor → Target**
	Military	Arrest some poaching suspects	MM0015	B → H
	Diplomatic	Petition to stop wildlife-caused crop destruction	D12719X3	F → A
Political	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 1: 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.

4.0.1 Acquiring EMAT action observations

Links between EMAT actions and actions reported in stories are discovered by running a parsing algorithm that is programmed within the **id** software system (see Supplement A). One of this algorithm's steps involves a search of each sentence in the story 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 m-word verbs: *arrested*, and *sentenced*. A similar partial match search is employed to to find direct object phrases, and prepositional phrases that (partially) match members of corresponding equivalence sets.

Action acquisition procedure

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

1. Acquire a file of raw HTML stories. The automatic system used here that scrapes stories from the web is described in Haas (2018).

2. Extract sentences, m-word verbs, direct object phrases, prepositional phrases, and EMAT action observations from these stories.

Step 2 consists of executing an *id language* command within the **id** software system Haas (2011, pp. 61-76). For example, the following **id** language fragment instructs the system to extract EMAT actions from a file containing stories about rhinos in Kruger National Park, South Africa:

```
report prepare_data
   parse_stories(kruger5417.txt false 1
      sarhinogroups.dat sarhinorgns.dat knp17prsd.dat knp17acts.dat)
```

In this language, `report` is a *main word*, `prepare_data` is a *qualifier*, and `parse_stories_()` is an *n-array assembly relation* (see Haas (2011, pp. 61-76)). The `parse_stories` relation, when run, executes the following four steps to extract a story's sentences, sentence components, and EMAT action observations.

1. Scan each story for the story's source.

2. 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.

3. 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. Off-the-shelf software for parsing HTML have been found to have 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.

4. Search within the tag-filtered story for its date, groups, actions, and regions. Perform these searches simultaneously within the Java program that implements this algorithm (see Haas (2017)).

   To search for actions, execute the following *EMAT action extraction algorithm*:

(a) Apply a parsing algorithm to the story's text fragment to search for m-word verbs, direct object phrases, and prepositional phrases. Use these to extract EMAT action observations. Parsing is accomplished with a modified version of the shallow parsing algorithm of Daelemans et al. (1999). Phrases are allowed to appear in any order within a sentence.

(b) For each sentence in the text fragment, compute an *overall similarity score* for each EMAT action by adding the similarity measure of the sentence's best-fitting m-word verb, best-fitting direct object phrase, and best-fitting prepositional phrase. See the next Section for a full description of this similarity measure. Then, for that sentence, create an EMAT action observation if this overall similarity score is greater than 1.9. Ignore actions for which the best-fitting m-word verb's similarity measure is less than 0.95 or the best-fitting direct object phrase's similarity measure is less than 0.95.

The EMAT action extraction algorithm of Step 4, above is a new version of an algorithm originally developed in Haas (2017). As the task of automatically detecting EMAT actions from online text can benefit from advances in computational linguistics, future improvements to this algorithm are likely. As soon as such a new algorithm is developed, the database should be re-created by re-running the method on the raw HTML stories.

A New algorithm for measuring phrase similarity

The algorithm for computing a measure of similarity between a member of an EMAT action equivalence set and a sentence component from a story is described below. This algorithm is a new version of an algorithm developed in Haas (2018). 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). Without loss of generality, let the shortest phrase be denoted ph_1, and the other phrase with ph_2. Let the number of words in the shortest phrase be $N = |ph_1|$. Then, a modified version of the measure given in Ponzetto and Strube (2007) is:

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

where s is the number of times n-gram pairs are formed by starting at the same location in each phrase, and m_n is 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. If both phrases are single words, $SIM(ph_1, ph_2)$ is 1.0 minus the Levenshtein distance between the two. This measure of phrase similarity is interpretable because it lies in the unit interval.

4.1 Learning

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 new 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 derived in-part, from a general outline for one given in Haas (2018).

4.1.1 Semi-Automatic Learning Algorithm

1. 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. List 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.

2. 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.

3. 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.

4.1.2 Example of discovering a new equivalence set member

The following sentence from the file `ef-script15-16.txt` (see Table **??**) 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.

4.1.3 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 2 should be added to the EMAT.

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

Table 2: A new EMAT action along with its initial equivalence set members.

5 Action extraction accuracy and speed

As discussed in Haas (2018), an algorithm designed to extract taxonomic actions from media

> 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*.

An assessment of the algorithm's accuracy was undertaken in Haas (2018). This study found that the fraction of actions correctly extracted was 87.5%, and the ratio of spurious actions to human-extracted actions was 0.125.

A timing study was also conducted in Haas (2018) where it was found that "on average, the algorithm processes a story in about 12 seconds on a personal computer running at 3.2 GHz." For example, about 12.5 hours would be needed to rerun the EMAT action extraction algorithm on the 3,724 raw HTML stories about rhinos in Kruger National Park.

References

Haas, T. C. *Improving Natural Resource Management: Ecological and Political Models*; Wiley-Blackwell: Chichester, U.K., 2011; 978-0-470-66113-0.

Haas, T. C. Automatic acquisition and sustainable use of political-ecological data. *Data Science* 2018, 17, p.17, DOI: https://doi.org/10.5334/dsj-2018-017 (accessed on 1 August 2018).

Haas, T. C. *Rhino Ecosystem Management Tool*, 2019. Available online: `www4.uwm.edu/people/haas/rhino_emt` (accessed on 2 July 2019).

Porter, M. F. An algorithm for suffix stripping. *Program*, 1980, 14(3), 130-137.

American Society for Indexing *Taxonomies & Controlled Vocabularies Special Interest Group*, 2018. Available online: `www.taxonomies-sig.org/about.htm` (accessed on 2 July 2018).

Yu, L. *A Developer's Guide to the Semantic Web,* 2nd ed.; Springer: Heidelberg, Germany, 2015; 978-3662437957.

McGuinness, D. L. 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, 2005; pp. 171-194; 978-0262562126.

van Rees, R. 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 2003. Available online: `http://itc.scix.net/cgi-bin/works/Show?w78-2003-432` (accessed on 2 July 2018).

Khan, S.; Safyan, M. Semantic matching in hierarchical ontologies. *Journal of King Saud University - Computer and Information Science* 2014, 26(3), 247-257.

Leng, R. J. *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, 1999. Available online: `www.icpsr.umich.edu/icpsrweb/ICPSR/studies/8606` (accessed on 2 July 2018).

Daelemans, W.; Buchholz, S.; Veenstra, J. Memory-Based Shallow Parsing. In *Proceedings of the EACL'99 Workshop on Computational Natural Language Learning (CoNLL-99)*, Bergen, Norway, 1999, 53-60.

Ponzetto, S. P.; Strube, M. Knowledge derived from Wikipedia for computing semantic relatedness. *Journal of Artificial Intelligence Research* 2007, 30, 181-212.

Levenshtein, A. Binary codes capable of correcting deletions, insertions and reversals. *Soviet Physics Doklady* 1966, 10(8), 707-710.

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