
AFRL-RI-RS-TR-2018-231

# TOWARDS LOCATING AND EXPLORING HARD-TO-FIND INFORMATION ON THE WEB

NEW YORK UNIVERSITY

*SEPTEMBER 2018*

FINAL TECHNICAL REPORT

***APPROVED FOR PUBLIC RELEASE; DISTRIBUTION UNLIMITED***

STINFO COPY

**AIR FORCE RESEARCH LABORATORY**
**INFORMATION DIRECTORATE**

■ **AIR FORCE MATERIEL COMMAND** ■ **UNITED STATES AIR FORCE** ■ **ROME, NY 13441**

# NOTICE AND SIGNATURE PAGE

Using Government drawings, specifications, or other data included in this document for any purpose other than Government procurement does not in any way obligate the U.S. Government. The fact that the Government formulated or supplied the drawings, specifications, or other data does not license the holder or any other person or corporation; or convey any rights or permission to manufacture, use, or sell any patented invention that may relate to them.

This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09. This report is available to the general public, including foreign nations. Copies may be obtained from the Defense Technical Information Center (DTIC) (http://www.dtic.mil).

AFRL-RI-RS-TR-2018-231 HAS BEEN REVIEWED AND IS APPROVED FOR PUBLICATION IN ACCORDANCE WITH ASSIGNED DISTRIBUTION STATEMENT.

FOR THE CHIEF ENGINEER:

**/ S /**
PETER ROCCI
Work Unit Manager

**/ S /**
TIMOTHY A. FARRELL
Deputy Chief, Information Intelligence
Systems & Analysis Division
Information Directorate

This report is published in the interest of scientific and technical information exchange, and its publication does not constitute the Government's approval or disapproval of its ideas or findings.

# REPORT DOCUMENTATION PAGE

*Form Approved*
**OMB No. 0704-0188**



| 1. REPORT DATE *(DD-MM-YYYY)* | 2. REPORT TYPE | 3. DATES COVERED *(From - To)* |
|---|---|---|
| SEPTEMBER 2018 | FINAL TECHNICAL REPORT | SEP 2014 – MAR 2018 |

**4. TITLE AND SUBTITLE**

TOWARDS LOCATING AND EXPLORING HARD-TO-FIND INFORMATION ON THE WEB

**5a. CONTRACT NUMBER**
FA8750-14-2-0236

**5b. GRANT NUMBER**
N/A

**5c. PROGRAM ELEMENT NUMBER**
62702E

**6. AUTHOR(S)**

Juliana Freire, Yamuna Krishnamurthy, Kien Pham, Aécio Santos, Sonia Quispe

**5d. PROJECT NUMBER**
MEMX

**5e. TASK NUMBER**
00

**5f. WORK UNIT NUMBER**
05

**7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)**
New York University
70 Washington Square S
New York, NY 10012-1019

**8. PERFORMING ORGANIZATION REPORT NUMBER**

**9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES)**

Air Force Research Laboratory/RIED
525 Brooks Road
Rome NY 13441-4505

**10. SPONSOR/MONITOR'S ACRONYM(S)**
AFRL/RI

**11. SPONSOR/MONITOR'S REPORT NUMBER**
AFRL-RI-RS-TR-2018-231

**12. DISTRIBUTION AVAILABILITY STATEMENT**
Approved for Public Release; Distribution Unlimited. This report is the result of contracted fundamental research deemed exempt from public affairs security and policy review in accordance with SAF/AQR memorandum dated 10 Dec 08 and AFRL/CA policy clarification memorandum dated 16 Jan 09

**13. SUPPLEMENTARY NOTES**

**14. ABSTRACT**

This work developed new methods and tools to empower subject matter experts to effectively discover and track information on the Web that is relevant to a given task (or domain). Our approach consists of two main components that address these challenges: 1) Domain discovery; and 2) Crawling and information gathering. For each of these components we have designed new methods, and developed open-source tools that implement these methods. Notably, we have designed a new framework that facilitates domain discovery, organization and presentation. We have also developed a general and extensible crawling infrastructure that substantially extends the ACHE open-source focused crawler to support complex crawling tasks and multiple crawling strategies to discover new content in a timely manner.

**15. SUBJECT TERMS**
Focused crawling, domain discovery, ACHE

| 16. SECURITY CLASSIFICATION OF: | | | 17. LIMITATION OF ABSTRACT | 18. NUMBER OF PAGES | 19a. NAME OF RESPONSIBLE PERSON |
|---|---|---|---|---|---|
| a. REPORT | b. ABSTRACT | c. THIS PAGE | | | PETER ROCCI |
| U | U | U | UU | 47 | 19b. TELEPHONE NUMBER (*Include area code*) N/A |

# Table of Contents

# List of Figures



# List of Tables

# 1 Summary

In this project, our main goal was to develop new methods and tools to empower subject matter experts to effectively discover and track information on the Web that is relevant to a given task (or domain). Our approach consists of two main components that address these challenges: 1) Domain discovery, which supports and guides users in the process of understanding how a given domain is represented on the Web to help them create a model for the domain; and 2) Crawling and information gathering, which given a search task and optionally a domain model, provides a scalable mechanism to continuously search the Web for information that belongs to the domain. For each of these components we have explored new research questions, designed new methods, and developed open-source tools that implement these methods. Notably, we have designed a new framework that facilitates domain discovery, organization and presentation, which enables users to seamlessly explore the content and create a computational model to recognize new content in the domain. This framework was implemented in the open-source Domain Discovery Tool (DDT) -- https://github.com/ViDA-NYU/domain_discovery_tool.

We have also developed a general and extensible crawling infrastructure. We have substantially extended the ACHE open-source focused crawler to support complex crawling tasks, multiple crawling strategies, as well as with the ability to efficiently re-crawl and discover new content in a timely manner. In addition, we have refactored the code and made the system more scalable and efficient, and improved its usability. These extensions have been released and are available at https://github.com/ViDA-NYU/ache.

The combination of ACHE and DDT provide a novel solution to discover and gather domain-specific Web information at scale, addressing key challenges set forth by the DARPA Memex program.

# 2 Introduction

The wide availability of data on the Web is a valuable asset for many applications. But it has also made it hard to find certain kinds of information. Search engines, such as Google and Bing, are the main entry points for users looking for information. They make use of massive computing power to both crawl the Web and create the search indexes, which currently cover hundreds of billions of documents. These systems, however, have limitations when faced with specific information needs. Because they aim to maximize coverage and breadth, queries often return a very large number of results, including many that are of little relevance. This leads to unnecessary information overload.

At the same time, due to resource limitations, search engines cannot download all the pages and documents on the Web and keep them up to date. As a result, pruning techniques are used and pages that might be important to a topic may be missed by a generic crawler. Similarly, while search engines schedule re-crawling to maintain their indexes fresh, information in certain sites or within a topic may become stale. Another limitation comes from the necessarily simple keyword-based interfaces provided. Such interfaces are not able to express structured queries over the markup and content, or analyses that require aggregation of data spread over multiple web pages or sites. For example, subject matter experts (SME) in an NGO who are trying to understand patterns of sex trafficking would like answers to questions such as ”What is the average price for escort service in New York?”, ”Given a physical description of an individual, how many escort ads in a given region match that physical description?”, “List all escort ads published today in NYC”. To answer these questions, they must first collect relevant pages. This is challenging for several reasons. First, while the SME may have an idea of the information she needs, this information may be represented in different ways across the Web. Thus, it is difficult to formulate a comprehensive set of search queries that retrieve the required information. And even when appropriate queries are formulated, the information stored by the search engine may be incomplete or outdated. Second, collecting this information manually is time-consuming, greatly limiting the coverage of the domain. This problem is compounded for tasks where information must be continuously tracked, especially when new sources of information are added constantly.

**Using Focused Crawlers to Find Domain-Specific Information on the Web.** Focused crawling has emerged as a scalable and effective mechanism to search for pages on a specific set of concepts that represent a small segment of the Web. Instead of attempting to cover all Web pages, a focused crawler tunes its search strategy to search for a target concept (or topic), while maximizing the number of on-topic pages it retrieves and minimizing the number of irrelevant pages visited. Focused crawlers thus bring many benefits: they lead to substantial savings in hardware and network resources compared to searching the *whole* Web; they make it cheaper to maintain the crawl up-to-date; crawls can go deeper and obtain a better coverage for a given topic; and the derived index, being focused, is more likely to return a higher fraction of actually relevant pages, reducing the information overload. While several focused crawling strategies have been proposed [17,18,19,20,21,22,23,23], there are still many barriers to their adoption, notably, configuring a focused crawler is challenging. To construct a model for a specific topic, the user needs to not only provide seed URLs that serve as the starting points for the crawl,

but also collect a set of positive and negative examples to train learning classifiers that recognize the target concept. In practice, these tasks are not only time consuming but they are also out of reach for users who are not familiar with computing methods, and in particular, machine learning. Surprisingly, these are problems that had received little attention in the literature.

Focused crawlers have been proposed to search for pages that belong to a given topic. However, many information foraging tasks require multiple search strategies to be employed. For example, once the SME from the NGO finds a set of sites that publish escort ads, she would like these sites to be crawled regularly so that they can quickly identify new ads. Furthermore, it would be useful for the crawler to find additional sites similar to those.

**Our Approach.** In this project, our main goal is to empower subject matter experts to effectively discover and track information on the Web that is relevant to a given task (or domain). Our approach consists of two main components that address these challenges: 1) Domain discovery, which supports and guides users in the process of understanding how a given domain is represented on the Web to help them create a model for the domain; and 2) Crawling and information gathering, which given a search task and optionally a domain model, provides a scalable mechanism to continuously search the Web for information that belongs to the domain. For each of these components we have explored new research questions, designed new methods, and developed open-source tools that implement these methods.

**Summary of Contributions.** We have identified and defined the problem of *domain discovery* search paradigm. We have designed a new framework that facilitates discovery, organization and presentation of domain specific content so the SME can explore and label Web content, and effectively translate her knowledge of the domain into a computational model. This is achieved by new methods that combine techniques from visualization, information retrieval, and machine learning. This framework was implemented in an open-source system called Domain Discovery Tool (DDT).

We have also developed a general and extensible crawling infrastructure. We used the ACHE open-source focused crawler as a starting point and substantially extended it to support complex crawling tasks as well as with the ability to efficiently re-crawl and discover new content in a timely manner. Based on the feedback from the SMEs, we have added a number of new features to make the crawler general and effective to a wide range of tasks and requirements. In addition, we have refactored the code and made the system more scalable and efficient. ACHE was integrated with DDT, allowing users to seamlessly go from exploring a domain and creating a model, to deploying a crawler to retrieve content.

The combination of ACHE and DDT provide a novel and usable solution to gather Web information at scale. In what follows, we describe both the research problems we addressed and the systems we built and released.

# 3 Methods, Assumptions, and Procedures

## 3.1 Domain Discovery: Domain Exploration and Modeling

Domain discovery is the process of acquiring, exploring, understanding and data in a specific domain. Example domains include human trafficking, illegal sale of weapons and micro-cap fraud. Before a subject matter expert (SME) starts the domain discovery process, she has an *idea* of what she is looking for based on prior knowledge. During domain discovery, as the SME obtains new content, she gains new knowledge about how the information she is looking for is represented on the Web. This iterative process is tedious and time consuming. A tool that streamlines domain discovery must cater to the following desiderata:

- Help SMEs learn about a domain and how (and where) it is represented on the Web,
- Simplify the process of labeling documents as relevant or irrelevant, and
- Acquire a sufficient number of Web pages that capture the SME's notion of the domain so that a computational model can be constructed to automatically recognize relevant content.

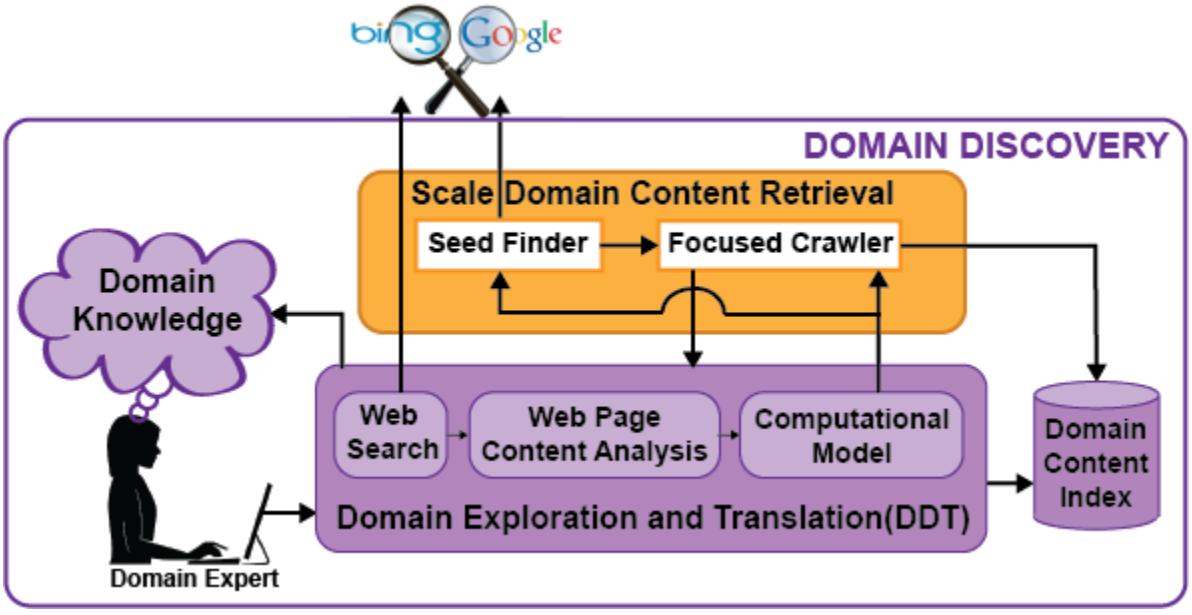



**Figure 1. Architecture of the Domain Discovery Tool**

Domain exploration and modeling is a core component of the domain discovery paradigm. We have used a Human-in-the-Loop (HITL) approach to this problem, and designed a framework, shown in **Figure 1**, that is intuitive, easy to use, and that guides users through the discovery process. Our design was informed by our interactions with SMEs and technical teams that participated in the program, and that represented several domains. We worked closely with agents and analysts from Alcohol, Tobacco and Firearms (ATF), analysts from Securities Exchange Commission (SEC), detectives working on human

trafficking cases, agents and analysts from Counterfeiting Intelligence Bureau (CIB). This allowed us to identify requirements and new research problems, as well as to develop solutions that have led to a new open-source.

### 3.1.1 Formalizing Domain Discovery

While acquiring knowledge about a domain, SMEs usually start with an idea about what that domain entails based on prior knowledge. However, this knowledge can be incomplete. Thus, the SME must search and explore the information on the Web so that she better understand how the domain is represented on the Web. This process is inherently iterative – as the SME obtains additional information, she incrementally learns more about the domain.

We have formalized and structured the human domain discovery problem by defining a set of operations that capture the essential tasks required to carry out the process. These operations can be grouped into four main classes:

- *Content acquisition:* SMEs must be able to discover new content. Thus, search operations must be supported that enable them to obtain and store these data at scale, ideally, automating tedious tasks. In addition, it should also be easy for them to input into the system their pre-existing domain knowledge.
- *Annotation*: As an SME explores content, it is important for her to record her findings. This can be achieved through annotations to the content. Besides marking pages as relevant and irrelevant to the target domain, it can also useful to add specific tags that provide additional information. For example, if an SME is exploring information about human trafficking, it may be useful to distinguish pages the contain ads from forums and new articles about the topic. These labels can later be used to build learning classifiers.
- *Summarization and organization*: Once pages are collected, it is important for the SME to have a high-level overview of the information. This can be achieved through techniques such as clustering and topic modeling. In addition, to help the use formulate queries that can retrieve additional, new pages, it is also useful to provide summaries of common terms and phrases that appear in the pages already in the collection.
- *Filtering and ranking*: As the volume of data increases, the SME must be able to search over the collected pages, and similar to search engines, the search results must be ranked to help the SME more quickly identify relevant information.

Examples of each operation type are given in **Table 1**.

***Table 1. Domain discovery operations***

| Operation Type | Operations |
|---|---|
| Content acquisition | Query the Web using a search engine (e.g., Google or Bing)<br>Upload known URLs<br>Crawl forward from a URL<br>Crawl backwards from a URL |
| Annotation | Label pages with custom tags<br>Labeling pages as relevant or irrelevant to the domain |
| Summarization and organization | Extract terms and phrases from content to summarize the pages<br>Discover topics in the pages through topic modeling<br>Cluster pages based on similarity |
| Filtering | Select a specific set of web pages using queries over the content and metadata of the pages, e.g., $'Ebola\ Symptoms'\ AND\ Tag\ =\ 'Relevant'$<br>Rank content by some criteria |

### 3.1.2 Creating a Domain Model

Domain definition and discovery can be viewed as an iterative process of mapping a SME's concept of the domain into a set of artifacts available on the Web (e.g., web sites, web pages, terms, phrases and topics). The artifacts produced are then used to instantiate a computational model, a function *f*, that is used to recognize documents that belong to the domain, i.e., the function $f$ determines whether an object is relevant or irrelevant to a domain $D$. Formally,

$$f(p)\ =\ s$$

where $p$ is a Web object (web page or any content identified by a URL) and $s\ \in [0,1]$ is a score assigned to $p$ which denotes the degree of relevance of $p$ to the domain. Using a threshold value $t$, we can determine the set of pages that constitute a domain $D$:

$$p \in D\ if\ f(p)\ >\ t$$

Ideally, given a domain $D$ and threshold *t,* there is a function $f_{oracle}$ that assigns accurate scores to all pages that exist on the Web. In practice, it is hard to build such a function. A domain discovery system should enable users to efficiently build a function *f* that approximates $f_{oracle}$ as well as possible, i.e., $f\ \approx f_{oracle}$ and encodes the SME's domain knowledge.

This can be achieved by:

1. Finding artifacts that support the construction a good function, such as:
    a. relevant and irrelevant web sites
    b. relevant and irrelevant web pages (i.e., any object identified by a URL)
    c. relevant and irrelevant terms (i.e., important terms for the domain such as: words, entities, tags, phone numbers, unique ids, etc.)
2. Enabling users to raise new questions/hypotheses about the domain and find additional, previously unknown artifacts related to the domain.

The operations discussed in Section 3.1.1 allow the user to fulfill these goals by mapping the user's knowledge to concrete artifacts. Given the size of the web and the potentially large number of artifacts, these operations should prune the search space, making it feasible for the user to find relevant information.

### 3.1.3 Organizing and Summarizing the Results of Operations

Section 3.1.1 provides operations for the user to gather, prune and select data from the Web to build the domain model. But in order to effectively use these operations the results of these operations should be presented in a user-friendly manner. The simple list of links with snippets provided by existing search engines is not sufficient for quick analysis and annotation of pages especially when the number of results returned are large. We propose to address this issue with richer interactive visualizations of the result pages such as multidimensional scaling (MDS) described in Section 3.4.

## 3.2 Domain Discovery API

We have implemented a Domain Discovery API[1] (DDAPI) that realizes the operations in Section 3.1.1 which acquire, annotate, summarize, organize, filter and rank content. DDAPI facilitates:

- The creation of different interfaces to satisfy different domain discovery needs
    - Once such interface is Domain Discovery Tool that we implemented using DDAPI (see Section 3.3 for details).
- Creation of different domain discovery workflows that mix and match the different operations, and which can be represented as (reproducible and configurable) scripts.

[1] http://domain-discovery-api.readthedocs.io/en/dd_api_docs/index.html

## 3.3 Domain Discovery Tool (DDT)

The Domain Discovery Tool (DDT)[2] is an interactive system that uses a human-in-the-loop-based approach to guide users in the domain discovery process (see **Figure 2**). It helps SMEs explore and better understand a domain (or topic) as it is represented on the Web by integrating human insights with machine computation (data mining and machine learning) through visualization. DDT allows an SME to visualize and analyze pages returned by a search engine or a crawler, and easily provide feedback about relevance. This feedback, in turn, is used to address three challenges:

- Assist SMEs in the process of domain understanding and discovery, guiding them to construct effective queries to be issued to a search engine to find additional relevant information;
- Provide an easy-to-use interface whereby SMEs can quickly provide feedback regarding the relevance of pages which can then be used to create learning classifiers for the domains of interest; and
- Support the configuration and deployment of focused crawlers that automatically and efficiently search the Web for additional pages on the topic. DDT allows users to quickly select crawling seeds as well as positive and negatives required to create the page classifier required for the focus topic.

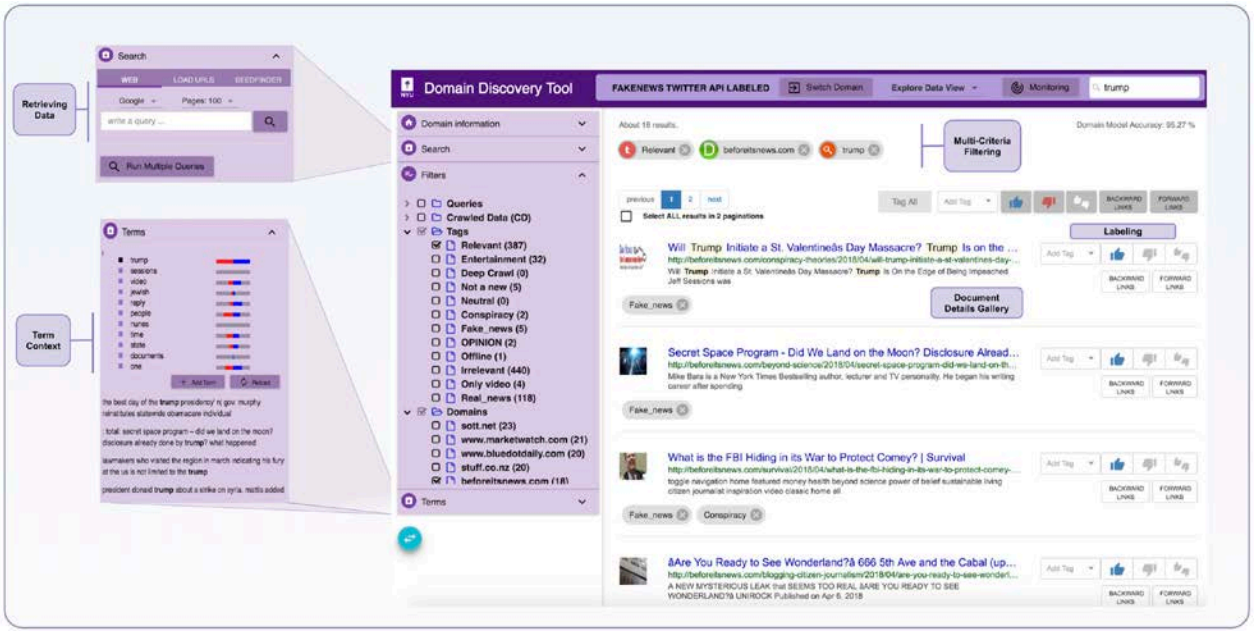



**Figure 2. User interface of the Domain Discovery Tool**

[2] http://domain-discovery-api.readthedocs.io/en/latest/

**Figure 2** shows a screenshot of the DDT user interface. An earlier version of the system was presented at the IDEA workshop at KDD [1]. The following subsections describe the key features of DDT.

### 3.3.1 Maintaining Search Context and Acquiring New Content

The search context, consisting of all pages/artifacts retrieved by the user, is stored in an Elastic Search[3] index. This allows the content to be queried and analyzed both during and after domain discovery. SMEs can use a variety of methods to make pages of interest available for analysis through DDT. These include:

**Querying the Web.** DDT allows users to query the Web using Google or Bing. They can leverage the large collections already crawled by the search engines to discover interesting pages across the Web using simple queries. Since search engines only return the URLs and associated snippet, DDT downloads the HTML content given the URLs and stores it in the selected domain index. This content can be used later for analysis of the domain and also as seeds for focused crawlers. Since downloading a large number of pages (including the raw HTML content) takes significant time, DDT performs this operation in the background.

**Uploading URLs.** In our interviews with experts and use cases we explored, experts often have a set of sites (or pages) they know are relevant. Therefore, it is important to provide a mechanism for incorporating this background knowledge. DDT allows users to provide URLs either through the input box provided or by uploading a file containing a list of URLs. DDT then downloads the pages corresponding to these URLs and makes them available through its interface.

**Crawling Forward and Backward.** DDT allows the retrieval of new pages by crawling a selected set of pages one level forward or backward. This allows expanding the collection with potentially relevant pages in a scalable fashion.

### 3.3.2 Summarizing and Organizing Results

Search engines display pages retrieved by their relevance to the search query, allowing users to view a subset of the pages at a time. DDT also supports paginated list view of ranked for the results for its filtering operations. However, this strategy is limited in that it does not provide an overview of all the results or allow the comparison of pages. DDT supports richer content organization and summarization through a new visual representation called Multiscale RadViZ, which is shown in **Figure 4** and described in Section 3.4.

[3] https://www.elastic.co/proucts/elasticsearch

### 3.3.2 Extracting Keywords

As shown in **Figure 2**, DDT presents to used relevant keywords and phrases that are extracted from the set of retrieved pages. The terms not only help the SME understand the existing documents, but also to discover new information about the domains of interest to refine their Web search or start new sub-topic searches. Search engines like Google and Bing suggested queries and query completions, but DDT also allows the user annotate the keywords and use these annotations to re-rank the terms to bring in more relevant keywords and phrases based on the user feedback.

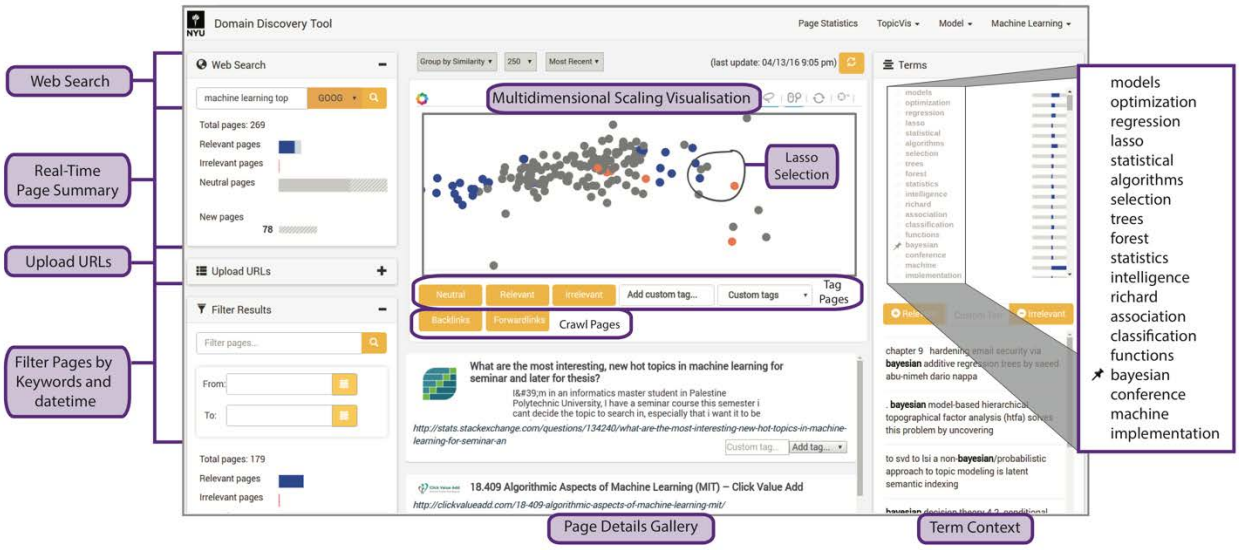



**Figure 3. Original DDT user inteface using the MDS visualization**

DDT ranks terms by their TFIDF [2] value to determine relevance. The higher the TFIDF the more relevant the term. Terms can be annotated as 'Positive' (blue) and 'Negative' (red). The tags are stored in the domain context. The term annotations are used to re-rank the unannotated terms using Bayesian Sets [3]. This brings the more relevant terms to the top which helps the user further discover the domain. Hovering the mouse over the terms in the Terms window displays the context in which they appear on the pages. This again helps the expert understand and disambiguate the relevant terms.

### 3.3.3 Annotating Content

Search engines do not provide a mechanism to annotate pages as relevant or irrelevant. DDT provides streamlined annotation capabilities for retrieved pages. The Multi-dimensional scaling (MDS), shown in  view provides a scaled visualization of the results which gives a sense of which documents are similar or dissimilar. It allows the user to

select a group of pages through a free-hand lasso selection, as in ***Figure 3***, which shows a screenshot of the original DDT user interface. The selection is used to display the page details, an image and text snippet from the page which helps the SME decide if the pages are relevant to the domain and tag them accordingly either as a group or individually.

DDT allows pre-defined tags ”Relevant”, ”Irrelevant” and ”Neutral”. It also allows SME’s create and assign their own Custom Tags to groups of pages. Relevance feedback through annotations is essential to:

- guide users in the process of understanding a domain; and
- configure focused crawlers [4] by using the feedback to build page classifier models and gather seed URLs.

### 3.3.4 Multi-Criteria Filtering

DDT allows filtering content by multiple criteria as shown in Figure 2. Filtering the pages of the domain by specified terms contained in the text of the pages, search queries, tags and date time range allows reducing the number of pages for analyzing and annotating.

### 3.3.5 Creating a Computational Model for the Domain

DDT helps users create a computational model that can recognize content that belongs to the domain, and thus enables the automated retrieval of data from the Web by focused crawlers. The annotations associated with the pages in the DDT index can be used to build a learning classifier. In the current version of the system, the annotations are exported to the ACHE [5] focused crawler, and ACHE creates the model. More specifically, it creates a Linear Support Vector Machine (LSVM) [6] classifier. DDT also exports a seed list that contains all the pages marked relevant. Using the seed list and the classifier, an ACHE crawl can be started that searches the Web for additional pages that belong to the target domain.
Note that the model is built once the user has completed the annotations. However, as the user annotates the pages she has no indication of the model accuracy or when she should stop annotating. To address this shortcoming DDT builds an online classifier model incrementally, as the user is annotating pages. The accuracy of this online model is made available to the user on the top right corner of the DDT interface as *‘Accuracy of Domain Model’*, as shown in Figure 2. The reported accuracy gives an indication of how good the model is helps user determine when to stop annotating pages.
The online classifier is a LSVM [6] with Stochastic Gradient Descent (SGD) [7] optimization. In order to not only predict the label but also get the probability of the class membership, we calibrated the model using Platt calibration [8] (since it works better with

smaller calibration sample) in order to decide which of the pages need to be presented to the expert for annotation that would improve the classifier model. This online model is then used to classify the currently unlabeled pages. Based on the label *L* and probability *P* of the class prediction and a given probability threshold *T*, the unlabeled pages are tagged:

- *"Maybe Relevant"* if *L* is "Relevant" and $P > T$
- *"Maybe Irrelevant"* if *L* is "Irrelevant" and $P > T$
- *"Unsure"* if $P < T$ irrespective of *L*

These labels provide additional information about the relevance of the pages and hence reduce annotation time by reducing the number of pages the user needs to analyze. The results of the evaluation of the online model, discussed in Section 4, show that annotating the pages tagged *"Unsure"* can lead to faster convergence of the model.

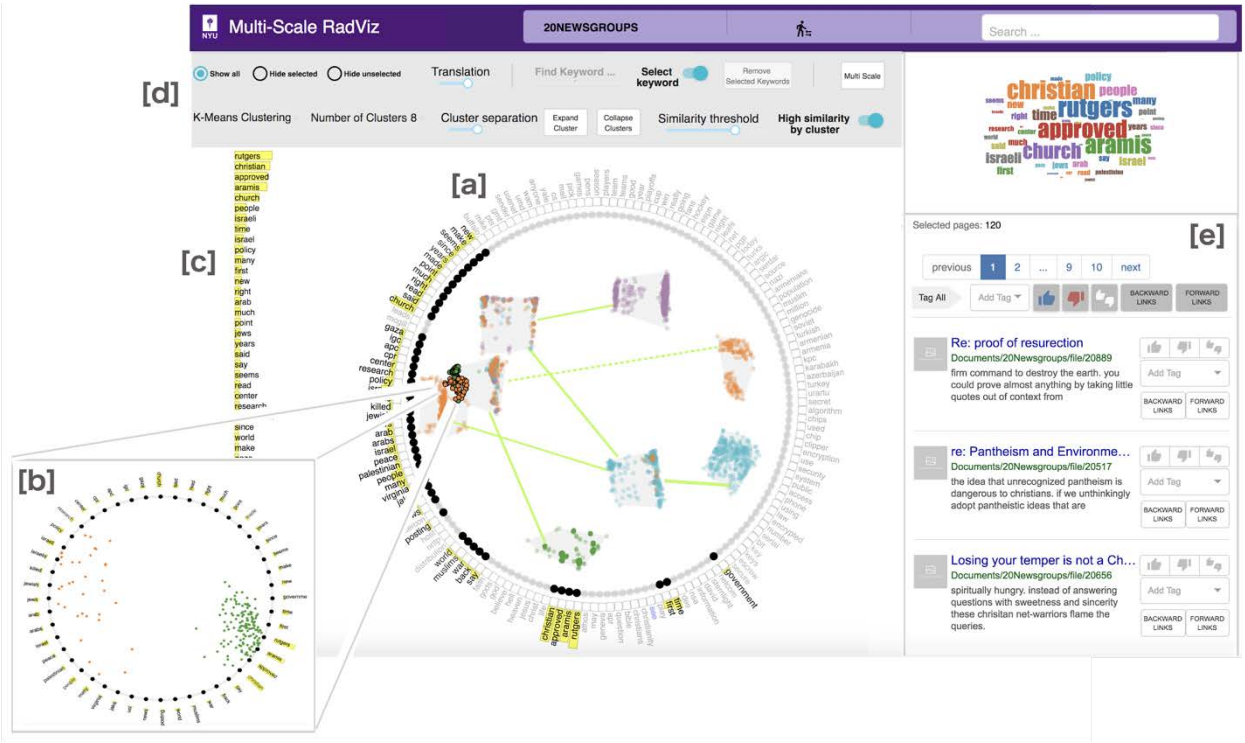



**Figure 4. Multi-Scale RadViz.** (a) MSR visualization showing cluster Level and information about their content. (b) Sub-MSR. (c) Control panel for interaction and setting MSR parameters. (d) Filtering by multiples Keywords. (e) Labeling and document detail gallery.

## 3.4 Document Explorer (DE)

We have developed a visualization system to explore and analyze text data called *Document Explorer (DE).* DE is an interactive framework that uses Multi-Scale RadViz (MSR), shown in **Figure 4**, which visualizes document collections at two-levels: document level and cluster level, providing meaningful overview, maintaining context during exploration and increasing detail upon demand. MSR supports visualization of projections while maintaining the context of the data: users can simultaneously visualize the attributes of the data and the data items. Hence, this visualization simplifies the identification of clusters, streamlines document labeling for learning a classifier, and helps users track how classifiers evolve as they are refined.

MSR initially visualizes documents at the cluster level. Keywords extracted from the selected pages are displayed around the circumference and its relevance is encoded as a yellow bar chart. When selecting a subset of documents, the relevance of keywords is updated. Furthermore, a sub-MSR (Figure 4(b)) can be generated using only the keywords related with the selected documents, allowing us to see clearer groups. Users can also go from cluster level to document level by clicking on the *'Expand Cluster'* button shown in **Figure 4**.

### 3.4.1 DE framework Overview

The following are the key features of the DE framework.

**Visualizing large number of documents.** Using MSR projection in DE, users can analyze and annotate groups of similar pages as opposed to each individual page, thereby considerably reducing the analysis (and labeling) time. The projection also shows information about the content of documents, therefore it is easy to infer relevance of these clusters of pages to the domain without examining the contents of the pages in a selected cluster. Additional features such as the word cloud, snippets and images of the selected web pages further enhance the understanding of the content. The DE framework also allows users to filter the documents by the anchor words.

**Exploring correlation between the vocabulary and the documents.** MSR also helps to identify the correlation between data items and attributes of the data. For text documents we use the most frequent words across the documents as the attributes. The MSR projection tightly couples the words to the documents that contain them. How close the document is to the word is an indication of how frequently that word occurs in that document. MSR also sorts the words chosen as anchors in the order of their semantic similarity. Furthermore, some interactions, such as sigmoidal weighting and rotate anchors, allow exploring the correlation of the attributes and seeing how they affect the clustering of the documents.

**Building and tracking the document classifier model.** MSR was extended to allow users annotate the pages as relevant or irrelevant to the domain in order to create a model that represents that domain. The anchors, word cloud, and snippets, help streamline the decision of relevance of the pages. Also, in order to help the user decide when they should stop annotating, and what pages should be annotated to improve model accuracy (with the least number of annotations), we build an online classifier model incrementally, while the user is labeling the pages. The accuracy of this model is made available to the user, serving as an indicator for the completeness of the model.

### 3.4.2 Feature Interactions

In this section we discuss the main interactions supported by DE framework (see Figure 4). Our goal was to support visual exploration, analysis and annotation tasks on document collections, allowing users to perform the following operations:

- **Summarization and detail on demand:** The system should provide an overview of large text collections, and enable their detailed exploration on demand.
- **Assisted labeling:** Users must be able to interactively label documents / groups of documents in the collection. The system should assist the users in the labeling process, by suggesting labels based on already labeled data.
- **Document relationships on demand:** The system should not only show the user how similar documents are, but also why this is the case. Details about the relationship should be displayed on demand.

Bellow we describe the interaction mechanisms supported by DE through MSR visualization which have been proposed to facilitate the visual analytics of data. We grouped these into three categories: data filtering, cluster interactions, and exploration of relationships between terms and documents.

**Data Filtering.** In order to enable a focus+detail exploration of the data, DE allows users to browse documents using two filter mechanisms: by keywords and by selection. After applying any of those mechanisms, the DE also allows users to transform the data in order to avoid clutter. The transformations are: 1) Show all data, 2) Hide selected data and 3) Hide unselected data. 'Show all data' projects all pages on MSR, however if users apply any filter, by keywords or by selection, all selected pages are highlighted and unselected pages are made almost transparent. On the other hand, 'hide unselected data' and 'hide selected data' act as expected: they hide data to allow user to focus her analysis on a subset of documents.

- *Highlighting by Keywords:* The main contribution of this mechanism is to enable the identification of documents strongly related to a keyword. Thus, by selecting one or more keywords from the MSR circle, documents in which those keywords are relevant will be highlighted. In addition, through this interaction, users are able to find new strong related keywords, since these documents are also related with other keywords which will be highlighted as well. Moreover, frequency bar charts associated with them will also be updated. For instance, given the keyword “hockey”, other keywords such as “game”, “team” or “players” could be suggested, giving a clearer context of the data to the user.

- *Filtering by selection:* Our framework allows the user to select a group of documents through a free-hand lasso selection. This feature can be used for multiple goals: (1) Interactively filter keywords: keywords related with the selected documents will be highlighted automatically and the bar charts associated with them with be updated to represent their relevance. (2) Interactive cluster analysis: users can expand selected clusters or create a second MSR view called sub-MSR (see Figure 3b). Sub-MRS is created by taking into consideration the related keywords from the selected documents. We explain sub-MSR in Section 3.4.2.2. (3) Update document details gallery: URL, images and text snippets from the selected documents will be updated automatically after selections.

**Cluster Interactions.** At the MSR cluster level, we present an overview of the data, by showing documents grouped into clusters. Using MSR, we can see the relationship between the clusters, clusters and keywords, and documents inside a cluster.

Below are detailed the main interactions for the exploration and analysis activities of clusters.

- *Expanding and collapsing selected clusters:* In MSR, there are two modes to display the documents inside the MSR circle: Expanded or collapsed. This option is available on the control panel. The objective of this feature is to allow users move from the cluster level to the document level, and vice versa in the same projection space. It depends directly on the type of analysis the user wants to perform. If the user wants to explore documents at a high level to get an overview of the data or see the relationship between documents in term of similarity, the collapsed view should be enough, since it shows the documents grouped in clusters. If the user wants to do a deeper analysis, they will move from the cluster level to the document level by expanding a cluster or documents at cluster level, selecting cluster or documents through a free-hand lasso selection, and clicking on the *'Expand Cluster'* button. At the document level users will be able to see a better correlation between documents and keywords since at document level the position of documents are directly defined by the frequency of keywords belonging to them.

- *Combined cluster and document level visualization:* The position of documents in the MSR circle is defined by the keywords displayed. In our visualization, there are some cases where using all keywords does not show how separate the documents are in the cluster, even at cluster level of MSR. In order to solve this problem, MSR can create a new RadViz projection we call sub-MSR. The new projection shows only the selected documents and the relevant keywords from these selected documents as shown in Figure 3b. Sub-RadViz is displayed outside MSR, allowing a combined analysis.

- *Interactive cluster separation:* Due to the large number of documents and limited space to display them, in some cases it is difficult to visually identify the separation among clusters. MSR provides a slider to control the inter-cluster separation between clusters. It allows users see how the clusters are separated more clearly. This functionality works like a zoom lens.

- *Similarity between clusters:* For labeling documents, getting an insight about similarity between clusters can help enormously, since it could guide the user to find or discard groups of documents more quickly. The position of clusters inside MSR is able to show certain kind of similarity, but cannot always represent semantic similarity since the position of clusters is determined by the position of keywords around the circumference. Therefore, it is possible that two clusters distant from each other can be semantically similar. To address this issue, MSR encodes the similarity between clusters as lines - we use cosine similarity since they are appropriate to work with document collections. Users can activate or

deactivate this option on demand. There are two views to show similarity information: (1) Similarity between all clusters, and (2) The highest similarity by cluster. Both use lines to represent the degree of similarity. The thicker the lines between clusters, the higher the similarity between them. However, each of them have a different mechanism of control. For (1), a threshold slider mechanism located at the control panel is used. Users can interactively show or hide lines by adjusting the threshold below or above its original value (similarity measure). Figure 3a presents the degree of similarity between 8 clusters. For (2), users can just show or hide the highest similarity by cluster. This view needs another mechanism of control since the highest similarity for some clusters could be low compared to the rest, thus, if the user uses the threshold slider mechanism of (1), the highest similarity of this cluster would be hidden. Showing the highest similarity by cluster is important since it can help identify clusters in which the data are similar to each other but not similar to other clusters. If a cluster is connected to many other clusters then it implies that the cluster has many documents that could belong to other clusters.

**Exploring Correlation between the Vocabulary and the Documents.** MSR also helps to identify the correlation between data items and attributes (keywords) of the data. The MSR projection tightly couples the words to the documents that contain them. How close the document lies to the word is an indication of how frequently that word occurs in that document. MSR also sorts the keywords chosen as DAs in the order of their semantic similarity. The following interactions allow exploring the correlation of the attributes and seeing how they affect the clustering of the documents.

- *Sigmoidal Weighting:* We use a filtering scheme based on sigmoidal weighting proposed by Ono et al. [9] to reduce cluttering and ambiguity in the visualization. Users can change the position of mapped keywords using an interactive control of sigmoid translation, a slider located on control panel called 'Translation', which can pull points towards the center of the circle or close to anchors based on the probability of a document to be more related with some keywords.

- *Rotate Anchors:* In order to discover the relationship between documents based on their anchors (keywords), users are allowed to change the anchor's position. If users think that some anchors could be related, they can position these anchors close to one another. When an anchor is relocated, we recompute the mapping of data points in MSR based on the new anchor's position. This can create interesting new clusters in the projection.

- *Remove Anchors:* MSR allows the removal of keywords. This feature does not only help to improve the visualization but also helps to discover new keywords from the document collection. When keywords are removed, new keywords are added automatically. In order to facilitate this process, a checkbox control is displayed next to each keyword. Users can activate or deactivate this mechanism of control. Figure 3d has this functionality activated.

## 3.5 Scaling Domain-Specific Content Discovery through Web Crawling

A particular domain may consist of a large number of pages, and thus, it is not feasible for a user to manually retrieve all these pages. Hence, there is a need to automate the retrieval of these pages to provide domain coverage. To solve this problem, we have substantially extended the ACHE focused web crawler ACHE [5]. Focused crawlers are effective tools to automatically retrieve large number of specific web pages relevant to a topic as determined by the domain model it is configured with. ACHE is able to collect web pages that satisfy some specific criteria, e.g., pages that belong to a given domain or that contain a user-specified pattern.

ACHE differs from generic web crawlers in the sense that it uses page classifiers to distinguish between relevant and irrelevant pages in a given domain. A page classifier can be defined as a simple regular expression (e.g., that matches every page that contains a specific word) or a machine-learning-based classification model. ACHE also automatically learns how to prioritize links in order to efficiently locate relevant content while avoiding the retrieval of irrelevant pages.

### 3.5.1 ACHE Improvements During DARPA's MEMEX Program

While ACHE was originally developed for research in focused crawling, based on the feedback we have received from the SMEs we collaborated with during the Memex program, we improved and extended it in several directions:

- We refactored and re-wrote nearly the whole code base to allow for easy development of new features.
- We fixed several stability problems in the software that caused random crashes during long crawls.
- We improved the performance of ACHE by developing a new multi-threading architecture to improve multi-core processor utilization. With this new design and by reducing the number of synchronization points in the code we improved parallelism and achieved an increase of 980% in the number of pages downloaded per second.

- We improved the software usability by creating a new configuration mechanism based on YAML files that include reasonable default settings for all parameters for the most common use cases and allows for easy configuration of other common use cases.

These improvements in ACHE allowed development of several new features, including the support for other domain-specific crawling tasks. ACHE now supports several features, which include:

- Crawling all pages of a fixed list of web sites (deep crawling)
- Discovery and crawling of new relevant web sites through automatic link prioritization (focused crawling)
- Classification of crawled pages using different types of pages classifiers based on machine-learning and regular expressions
- Crawling hard-to-find pages from "dark web" hidden services in the TOR network by using TOR proxies
- Crawling of password-protected web sites by allowing configuration of authorization cookies retrieved from user browsing sessions
- Continuous re-crawling of pages in order to discover new pages
- Configurable data output to popular state-of-the-art data-management systems such Elastic Search and Apache Kafka
- A web interface for system monitoring and searching the crawled pages in real-time
- Integration with an external monitoring system (Prometheus)
- A REST API for crawler monitoring and management
- Support for running multiple crawls in the same crawler instance (multi-tenancy)
- Exact and near-duplicate pages detection

### 3.5.2 ACHE Crawling Strategies

ACHE supports different crawling strategies, including:

- Focused crawling: crawls the Web in search of pages that belong to a given topic (or domain) as represented by a learning classifier
- In-Depth Web Site Crawl: Given a list of URLs (sites), ACHE will crawl all pages in each site. The crawler stops when no more links are found in the sites.
- Dark-Web crawling: crawls sites that reside on the Dark-Web using the TOR protocol

The system has several configuration options to control the crawling strategy, i.e., which links the crawler should follow and how priority is assigned to each link.

**Scope.** Scope refers to the ability of the crawler only follow links that point to the same "host". If the crawler is configured to use the "seed scope", it will only follow links that belong to the same host of the URLs included in the seeds file. For example, if the scope is enabled and the seed file contains the following URLs:
http://pt.wikipedia.org/
http://en.wikipedia.org/
then the crawler will only follow links within these two domains. Links to any other domains are ignored.

**Hard and Soft Focus.** focus mode (hard vs. soft) is another way to prune the search space, i.e., discard links that will not lead to relevant pages. Relevant pages tend to cluster in connected components, therefore the crawler can ignore all links from irrelevant pages to reduce the amount of links that should be considered for crawling. In "hard-focus mode", the crawler ignores all links from irrelevant pages. In "soft-focus mode", the crawler does not ignore links from irrelevant pages, and relies solely on the link classifier to define which links should be followed and their priority. When the hard focus mode is disabled, the number of discovered links grows quickly, so the use of a link classifier (described below) is highly recommended to define the priority that links should be crawled.

**Link Classifier.** The order in which pages are crawled depends on the link classifier used. A link classifier assigns a score (a double value) to each link discovered, and the crawler crawls every link with a positive score with priority proportional to its score. The *max depth link classifier* assigns scores to discovered links proportional to their depth the web tree (assuming the URLs provided as seeds are the roots) and will ignore any links whose depth is higher than the configured threshold.

### 3.5.3 Crawling the Dark Web

TOR is a well-known software that enables anonymous communications. "Dark Web" sites which reside on the TOR network are usually not crawled by generic crawlers because the web servers are usually hidden and can only be accessed through specific TOR-based protocols. Sites on the TOR network are accessed via domain addresses under the top-level domain ".onion". To crawl such sites, ACHE relies on external HTTP proxies, such as Privoxy, configured to route traffic through the TOR network. To have ACHE crawl dark sites, besides configuring the proxy, users can configure ACHE to route requests to ".onion" addresses via the TOR proxy.

### 3.5.4 SeedFinder

ACHE includes a tool called SeedFinder, which uses search engines to discover additional pages and web sites that contain relevant content. After you have your target

page classifier ready, you can use SeedFinder to automatically discover a large set of seed URLs to start a crawl. You can feed SeedFinder with your page classifier and a initial search engine query, and SeedFinder will automatically generate queries that will retrieve additional relevant pages and issues them to a search engine until the max number of queries parameter is reached. SeedFinder is available as an ACHE sub-command.

## 3.6 Bootstrapping Domain-Specific Content Discovery with Minimal User Feedback

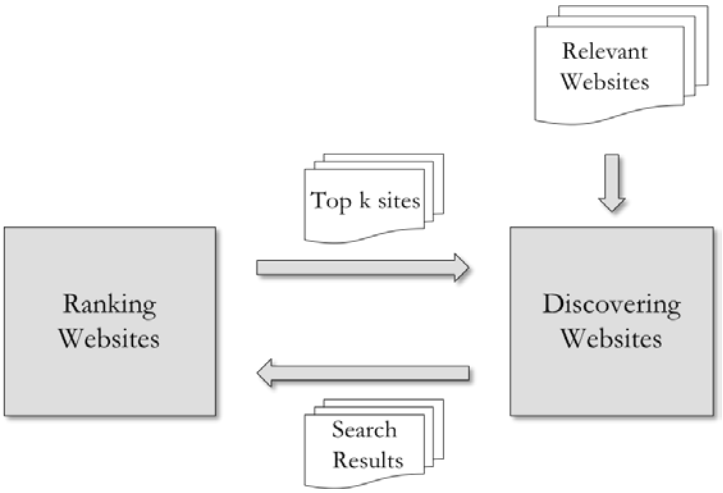



**Figure 5. Web Site Discovery Framework**

As previously mentioned, the domain discovery process has been traditionally carried by SMEs manually, using web search engines such as Google and Bing. Although the Domain Discovery Tool aims to streamline this process using a human-in-the-loop approach, we also investigated to what extent the domain discovery process can be completely automated and carried out with minimal user input and feedback. To address this challenge, we developed a ranking-based framework that iteratively discovers a large number of relevant websites given only a very small set of relevant sites as input.

***Figure 5*** presents an overview of the framework that consists of two components: web site ranking and web site search. The initial set of domains is usually provided by SMEs as seeds. Each following iteration proceeds by first expanding the known set of domains using search operators (e.g., keyword search and backlink search) and then selecting additional relevant domains to be used as seeds for the next iteration. The key idea behind the framework is to employ similarity-based ranking techniques to automatically select content that has a high probability of being relevant, and then use it as a form of pseudo-relevance feedback (i.e., replacing human feedback by automatically generated

feedback) to further expand the initial set of relevant domains. By using ranking, we reduce the selection of false positive content selected at each iteration and therefore increase the overall harvest rate.

## 3.7 Timely Discovering Domain-Specific Content

For some applications, such as human trafficking, it is critical to discover new content in a timely fashion. We explored on the problem of timely discovery of new content in a domain-specific setting. More formally, we define the content discovery problem as follows: given a set of seed pages *S* and we want to select the top-k pages $S_k^t$ for every timestamp *t*, where $S_k^t \subset S$ and $|S_k^t| \ll |S|$ , such that re-crawling every $S_k^t$ at a timestamp *t* maximizes the number of new (relevant) links discovered. We assume that new content can be discovered by re-crawling previously crawled pages, but instead of crawling them periodically using a fixed schedule, we propose new algorithms that learn and leverage page change patterns to dynamically *derive efficient re-crawl schedules that optimize the new content discovery rate over time*. Unlike previous approaches which assumed the crawler has full knowledge of how pages change over time [10, 11], we dynamically learn them as the crawl proceeds and more knowledge about pages is acquired.

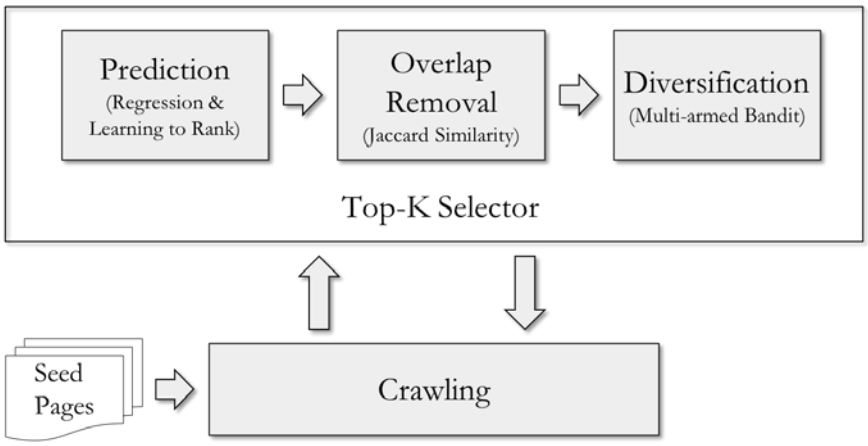



**Figure 6. Domain-specific content discover framework**

We proposed a two-stage framework to generate dynamic re-crawl schedules. The framework is illustrated in the **Figure 6**. In the first phase, we predict the number of new outlinks $\hat{o}$ that the page will yield at time *t+1* and use the prediction to select a set of candidate pages. To do so, we identify a set of useful features that are good predictors for pages that lead to a high yield (i.e., are likely to contain links to new pages) and use machine-learning based algorithms that combine these features to estimate $\hat{o}$. Because different pages may share links, selecting pages that have high yield but whose link sets overlap would negatively impact the overall performance. Thus, during the second phase,

we rank the candidate pages taking into account not only the estimated number of new links, but also the estimated overlap among the sets of outlinks in the associated pages.

While the greedy, learning-based approach is effective, it may suffer from bias: by selecting only pages that are expected to have high-yield, it may miss new pages could lead to higher yields. A common approach to this problem is to introduce exploration: instead of just exploiting known seeds, we can explore to find (and learn the patterns from) new seeds. The challenge is how balance exploration and exploitation. We proposed a method that uses the multi-armed bandits strategy [12] to automatically select an exploration threshold and to dynamically adapt the threshold as the crawl progresses.

# 4 Results and Discussion

## 4.1 Domain Discovery Tool

DDT has been release as open-source (Apache License). The system can be downloaded from https://github.com/ViDA-NYU/domain_discovery_tool and its documentation is available at http://domain-discovery-tool.readthedocs.io/en/latest.

We conducted experiments to evaluate the different components of our domain specific search paradigm. We start with the evaluation of the domain exploration and translation component implemented by DDT. The evaluations include:

- A user evaluation that demonstrates the overall usefulness and ease of use of DDT in exploring and gathering the domain specific Web pages as compared to existing Web search engines, and
- A simulation that shows how faster convergence to a good model can be achieved by using the online learning approach.

### 4.1.1 Surrogate User Group Evaluation

DDT was evaluated by the surrogate user group (SUG) from Sep 8 - Sep 14, 2015. This evaluation was done with an older implementation of DDT. The goal of the testing was to evaluate the effectiveness of DDT's functionalities in the context of the following two work processes:

- Orientation in a new domain
- Discovering new information and sources relevant for investigation in a new domain

The test began with a live demo of DDT, to the participants, via desktop sharing in Google Hangout. The participants were also provided with a detailed written report of the features and functionalities of DDT, demo videos and scripts. During the whole testing phase we communicated with each other through emails. The details of the test and the feedback provided are as follows:

**Participants.** DDT was tested by 3 members of the SUG. Two of the participants were an information systems technologist and a data scout from IST Research[4]. The third participant was the COO of Marinus Analytics[5]. Each of them explored one or more domains using DDT. The domains explored were *affiliate marketing fraud, domestic labor, police brutality, and cyber bullying*.

The participants were given a feedback questionnaire, that would help us better understand the effectiveness of our various design choices in DDT. At the end of the evaluation each participant provided us the duly filled feedback questionnaire. In addition they also provided us with a consolidated review of DDT that summarized their individual experiences with the tool.

**DDT Evaluation Setup.** We made DDT tool available via the Web to the SUG. The DDT server and the elasticsearch instance ran on an AWS server.

**Evaluation Feedback.** We received both positive and negative feedback of the design and workflow from the SUG. They are discussed below.

*Quality of Features.* Some of the feedback about the features of DDT were as follows:

- The users noted that page previews were typically enough to easily distinguish between relevant and irrelevant pages.
- They liked the ability to select a group of pages with the free-hand lasso selection in the MDS visualization window to annotate the pages. But the response to the MDS visualization was mixed. We concluded that we should provide a better understanding of the MDS visualization to end users. We redesigned the MDS to the MSR, described in Section 3.4, based on this feedback.
    - One user said *"Fundamentally, I was impressed with the tool because you can quickly assess internet search results over traditional pagination*

[4] http://istresearch.com/
[5] http://www.marinusanalytics.com/

*approaches using the visual clustering method. Quickly rolling over pages is enlightening to the content which exists online. With huge number of results, it helps to draw your attention to outliers or core members of a cluster which is effective in sifting through the returned pages."*

    - However, another user had a few questions *"Should I choose things clustered closely together? No? Why are they clustered closely together when I think that a more distant node seems to be very similar to some of them? The location of page 'nodes' had little to do with the pages' relevance in my experience. I did not feel it was extremely important to understand the clustering process, however"*.
- They found the positive and negative bars in the extracted *Terms* window informative and helpful in choosing the relevant and irrelevant terms. To quote one of the users *"I thought it was pretty neat"*. Another user said *"This seemed like a potentially interesting step, the bars making it visually clear what terms were appearing in the selected relevant/irrelevant pages."*

*Quality of Findings.* The following are the observations of the SUG about the quality of findings with the tool.

- The users reported finding new information during their exploration of domains with DDT. In one SUG user's demo domain (cyber bullying), she did not know how to find the *"community of evil"*. The tool successfully introduced:
    - People talking on YouTube about how to bully online
    - People posting videos potentially calling for others to be bullied and harassed
    - News articles discussing cyber bullying and detailing past instances
- Another user noted *"This helped me identify blogs for suspected affiliate marketing scammers. I did not know these blogs existed. This was a new find and very relevant for tracking crime."*
- The users reported that results got better over time with annotations. To quote one of the users, *"The searches definitely got better over time. I think the recall was still relatively low, however, given my experience with Google searching."*
- The users unanimously found the quality of terms extracted was very good. They reported that the extracted terms got better with annotations. To quote one of the users in the context of the cyber bullying domain, *"Yes, I did learn new terms specific to the types of attacks like 'doxing'."*

*Feedback Summary.* The features the SUG most liked about the tool were:

- Rapid search term feedback with improved results following annotated and custom terms
- Ability to annotate and analyze groups of pages
- Page and term previews provided

The features they had trouble with were:

- The workflow was not very clear. They were unsure of the sequence of steps to follow from making the query to iteratively improving their results
- The feedback of how their annotations were actually helping was not explicit
- Not being able to explicitly look at the history of their annotations and queries

One of the users noted that *"All of the functions present have the potential to be very useful once molded into a more user friendly/understandable framework."*

The feedback provided us a detailed understanding of the pros and cons of our framework. It helped us make design a completely new user interface (available in the current release of the system) which addressed most of the issues raised.

### 4.1.2 User Evaluation of DDT

As an initial validation for our design decisions, we carried out a small-scale study. Since search engines are the most common tool used for gathering information on the Web, our study compares the effectiveness of DDT with that of Google for gathering information in a specific domain.

**Experimental Setup.** The evaluation involved six participants. The participants were graduate students or research associates with background in computer science. The two primary criteria for their selection was (1) that they should be very familiar with using search engines, especially Google and Bing, and (2) they should be capable of exploring information about a given topic on the Web. The users were given a demo of DDT and all its features, and they were allowed to use DDT to get familiar with it before the actual evaluation. In order to keep the topics easy to understand and to ensure that the participants were not experts in the domain (as the goal here is for them to discover pages for the given topics ), we selected topics from the Ebola domain in the TREC Dynamic

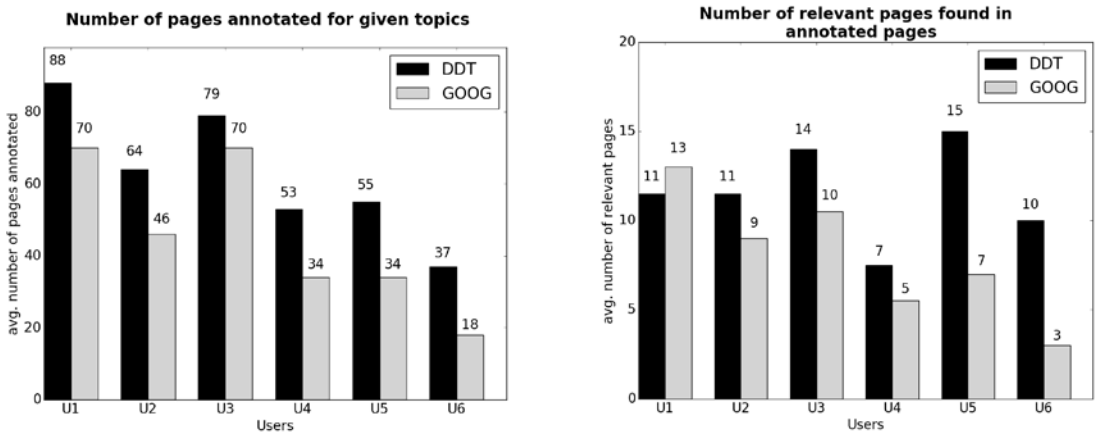
Domain (DD) Track 2015 dataset[6]. The dataset for the Ebola domain consists of $\approx 143{,}044$pages of which $\approx 5{,}832$pages are labeled by humans into 20 topics.

Each user was then given the same 2 topics (since the data is part of TREC DD evaluation we are not allowed to disclose the topics), in the Ebola domain, and asked to .nd as many pages as they could for each of those topics using Google and DDT. While using Google the users annotated pages relevant to each topic by bookmarking them under corresponding folders. For DDT, the users annotated the pages for each topic with a custom tag corresponding to that topic. They were allowed 15 minutes for each topic on Google and DDT.

Since we used Google as a search engine in our experiments, we needed a "domain expert" that could consistently judge whether a page annotated by a user belonged to the given topic or not. Since we did not have access to such an expert directly, we instead built a multiclass SVM classifier, using the TREC DD data that was labeled by humans. The words (excluding stopwords) in the pages were used as features and the topic a page belonged to was the output class. The model was tested using cross validation which produced an average accuracy of 74.6%. Given the topic distribution, where the most frequent topic consisted of 700 pages, the model is still quite good, as a max baseline accuracy, if we labeled all samples with the most frequent topic label, would be $(700/5832) * 100 = 12\% << 74.6\%$.

**Results.** We measured the total number of pages that the users were able to annotate with Google and DDT. We executed the model on the annotated pages to find how many of them were actually relevant to the given topics. The results are shown in ***Figure 7***.

[6] http://trec-dd.org/2015dataset.html




(a) (b)

**Figure 7. Evaluation: Comparing Google and DDT for domain discovery**

***Figure 7 (a)*** plots the average number of pages annotated for the topics by each user. Users were able to annotate more pages using DDT than Google. Users reported that visualization and grouping of the pages by similarity made it easier for them to select and annotate a set of pages. Whereas on Google, they had to go through the list of results on multiple pages to be able to find the relevant pages and then bookmark them individually. Figure 6b shows the average number of relevant pages found by each user. The plot shows that the majority of the users were able to find more relevant pages with DDT than Google – in some cases 2-3 times more pages. .is indicates that the features provided by DDT do help streamline domain discovery. .e only exception was user U1. This user used the least number of features of DDT, which could explain the lower relevant pages found.

**User Feedback.** Users also completed a questionnaire about their experience with DDT. The following are the summarized positive and negative feedback we received. Given the duration of 15 minutes for each topic the users were not able to use all the features of DDT. The union of the set of features used by each user for this experiment were web search, MDS visualization window, backlinks and forward links, various filtering options and page tagging.

*Positive:*

- The users found the MDS visualization of the pages useful to see the similarity between the pages, analyze and annotate a group of pages
- The various methods to filter pages, such as by queries, tags and “more like this” (pages similar to a selected set of pages), facilitated finding and bringing in more pages related to the domain for analysis
- Ability to add user defined tags to annotate a set of pages allowed grouping them by topic

- Avoiding annotating the same pages multiple times as they are brought in through different queries
- Though none of the users was able to use the terms extracted due to the limited time of the test, the consensus was that the extracted terms were relevant to the domain and improved with page annotations

*Negative:*

- The feature for crawling forward and backwards from selected pages was difficult to use and led to a large number of irrelevant pages. .is was especially true for the Ebola domain as most of the pages for this domain were news articles with links to different unrelated topics
- Although DDT was easy to use with li.le training, some aspects like the need for tagging extracted terms, the workflow (the sequence in which data gathering and analysis should be done) were not clear.

### 4.1.3 Online Classifier Performance Results

We set up an experiment to evaluate the benefit of the online model. The dataset used was from the human trafficking domain annotated by the NYU team. It contained 446 relevant pages which were any page that contained escort ads or erotic massage parlors. They were annotated based on the questions provided during the DD evaluation. The dataset also contained 301 irrelevant pages.

The experiment was a simulation of a user annotating pages with and without the feedback from an online model learner. We did 5 iterations where each iteration consists of 20 sets of annotations. The feedback is the pages that the model is unsure about. So with feedback the simulation annotates only pages that the model is unsure of in each of the 20 runs. Without the feedback the simulation annotates randomly the unlabeled pages in each of the 20 runs. After annotations in each run the online model is updated and the accuracy of the model is computed.

After each run the online model is updated and used to label the not yet annotated or unlabeled pages as described in Section 4.5. The feedback is the pages that the model is unsure about. So with feedback the simulation annotates only pages that the model is unsure about in each of the 20 runs. Without the feedback the simulation annotates randomly the unlabeled pages in each of the 20 runs. A.er annotations in each run the online model is updated and the accuracy of the model is computed by cross-validation.

The results of the accuracy per run for the 5 iterations are shown in ***Figure 8***.

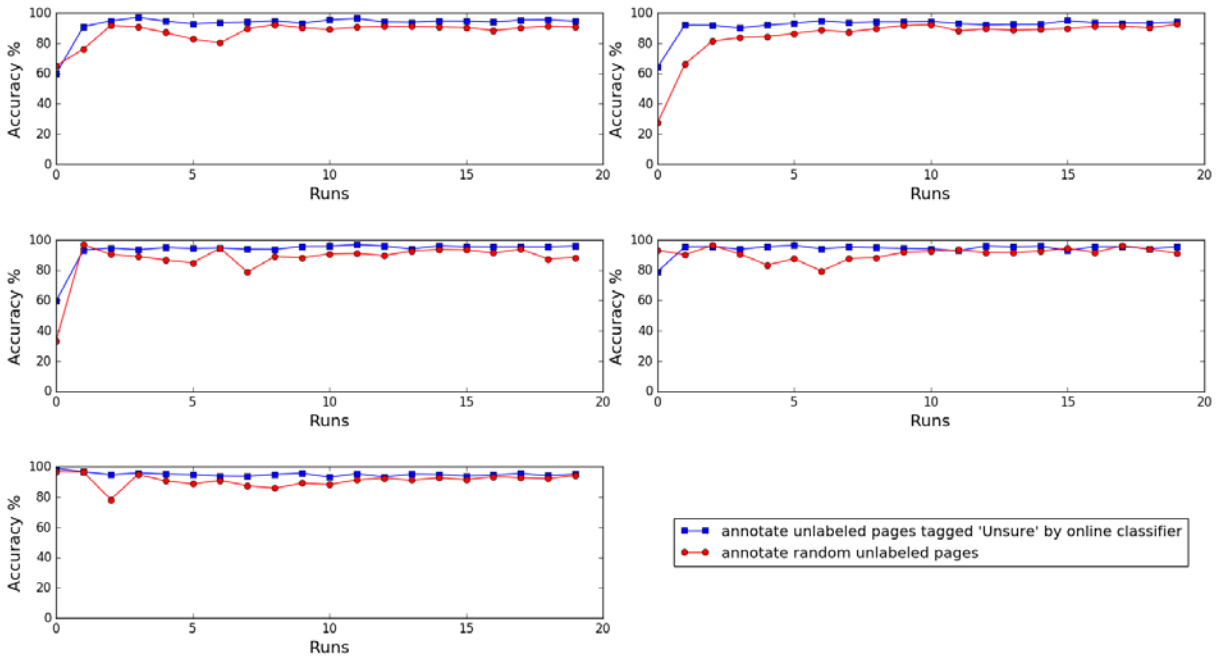



**Figure 8. Model accuracy for 5 iterations each with 20 sets of annotations**

From ***Figure 8*** we can see that:

- For each iteration the accuracy of the model with feedback is overall better than that without feedback; and
- The model converges more quickly (that is, with fewer annotations) to acceptable accuracy with feedback than that without feedback.

Hence, we see that the online model is better as it not only provides an indication of the accuracy of the model but also helps the user reduce the number of required annotations.

## 4.2 Bootstrapping Domain-Specific Content Discovery with Minimal User Feedback

We conducted extensive experiment in the three different domains: weapon marketplace, weapon forum and human trafficking (escort), and show the effectiveness of our approach over state-of-the-art methods. We are also the first to compare the performance of different search operators on the considering domains.

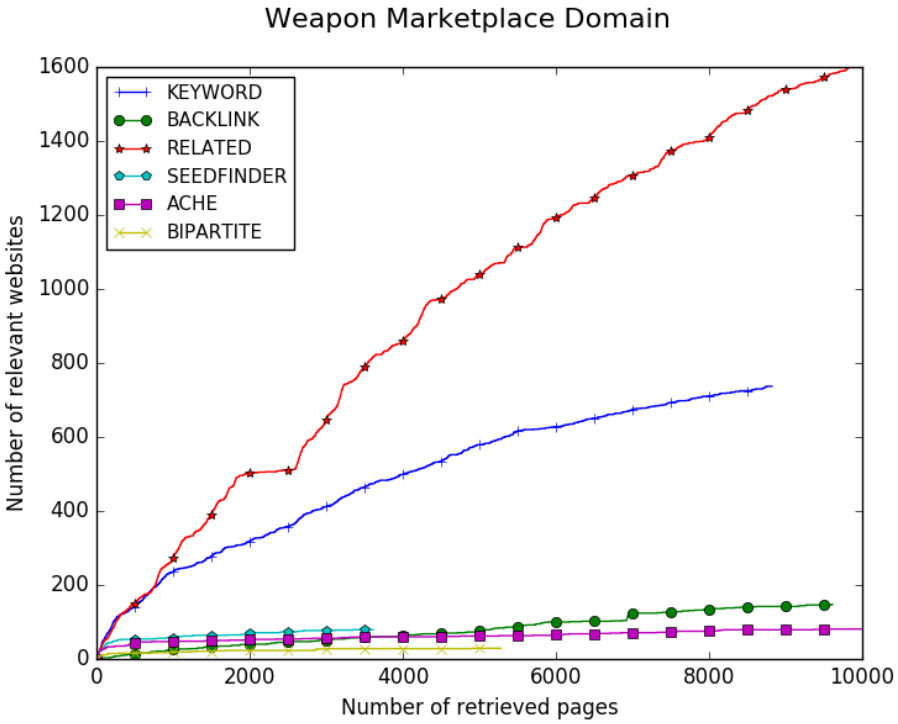



**Figure 9. Comparison of Coverage between Baselines and Proposed Methods**

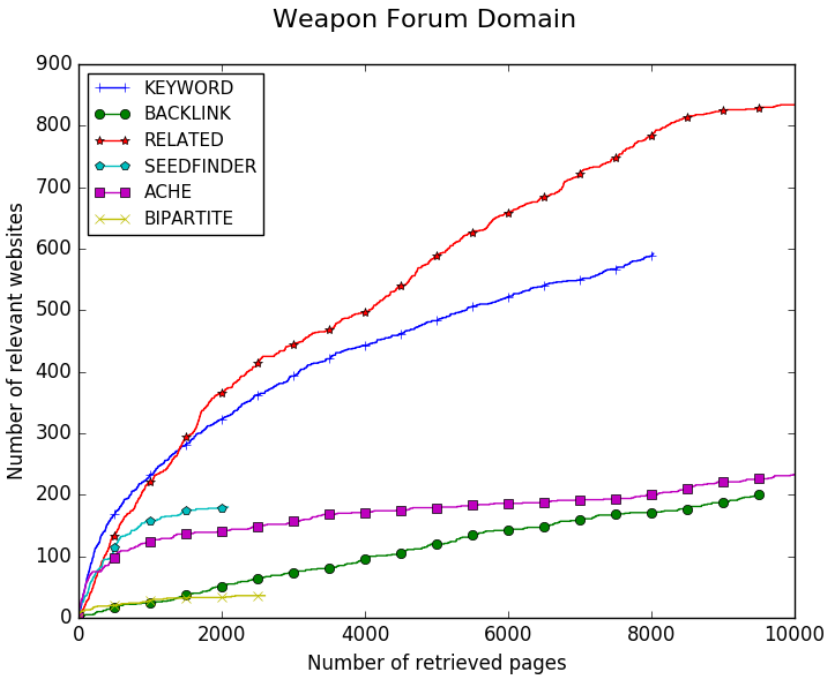



**Figure 10. Comparison of Coverage between Baselines and Proposed Methods**

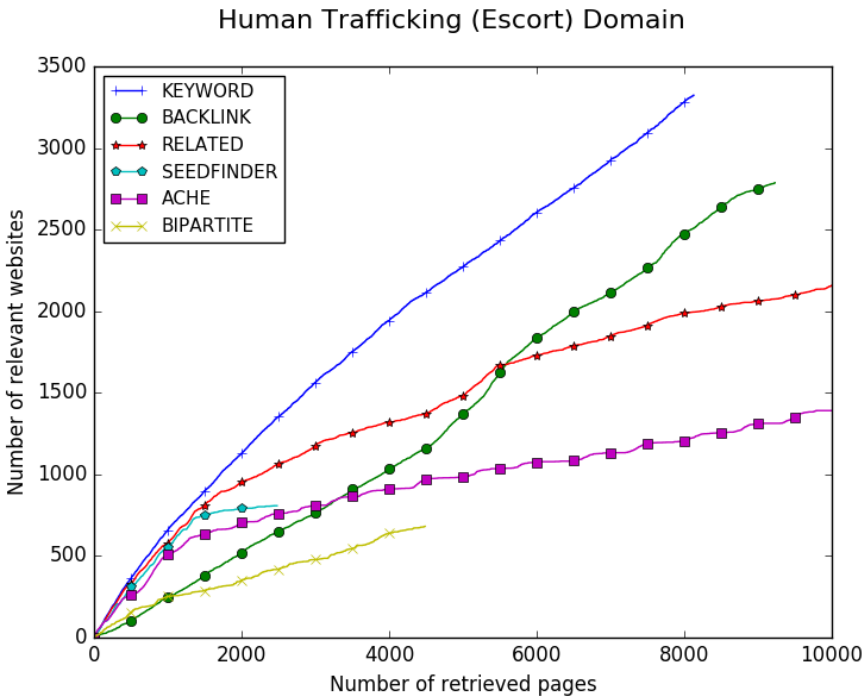



**Figure 11. Comparison of Coverage between Baselines and Proposed Methods**

Figures 9, 10 and 11 present the harvest rate (number of relevant websites discovered versus number of visited pages) attained by our proposed methods and the baselines. Our framework using keyword search, related search and backlink search are denoted as keyword, related and backlink respectively. Seedfinder [13], ACHE [5] and bipartite [14] are different discovery strategies introduced in the literature. The results clearly show that our proposed framework using keyword search and related search outperform other baselines in the three considering domains.

## 4.3 Timely Discovery of Domain-Specific Content

To assess the effectiveness and efficiency of the proposed approach, we performed a detailed experimental evaluation to compare it against state-of-the-art discovery techniques using real data from different domains. The results show that the features we selected and the strategy used to combine them lead to effective predictors of page yield. By learning these features and taking the overlap into account, our algorithm outperforms all other approaches: it achieves higher coverage using less resources. In addition, by balancing exploration and exploitation, higher coverage is attained. Figures 12, 13 and 14 show how coverage varies over time for the different strategies. Although the magnitude of the difference between these methods seem to be smaller, our proposed methods (i.e., REG, REG-RR, BANDIT and LTR) still outperform all the baselines [11].

We refer to our paper [15] for more details about the proposed framework and the experiment.

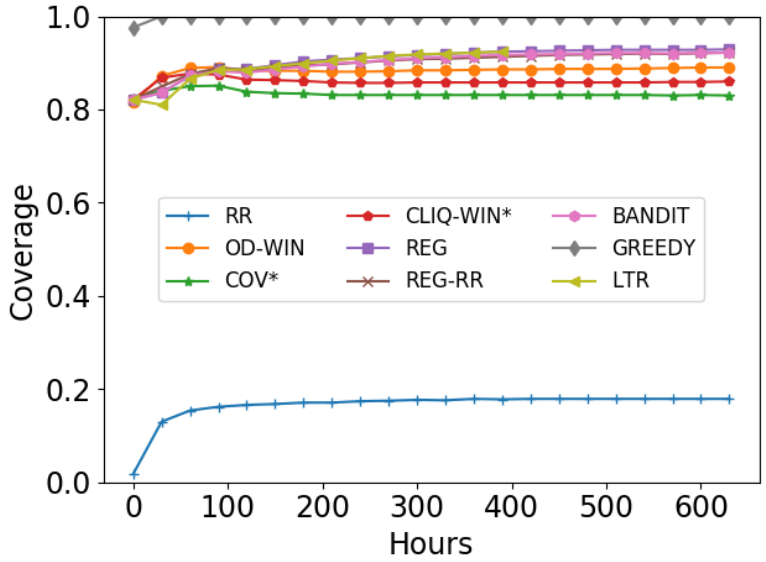



**Figure 12. Comparison of Coverage between Baselines and Proposed Methods for the Human Trafficking (Escort) Domain**

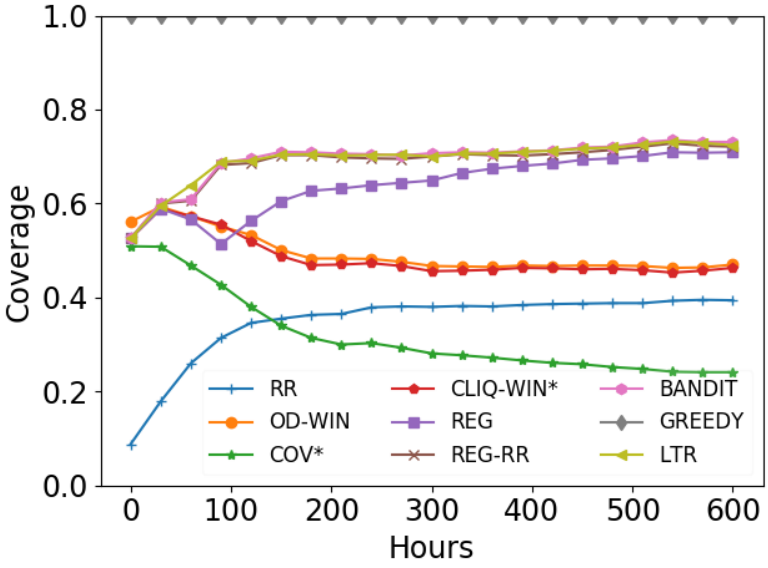



**Figure 13. Comparison of Coverage between Baselines and Proposed Methods for the Humanitarian Crisis Domain**

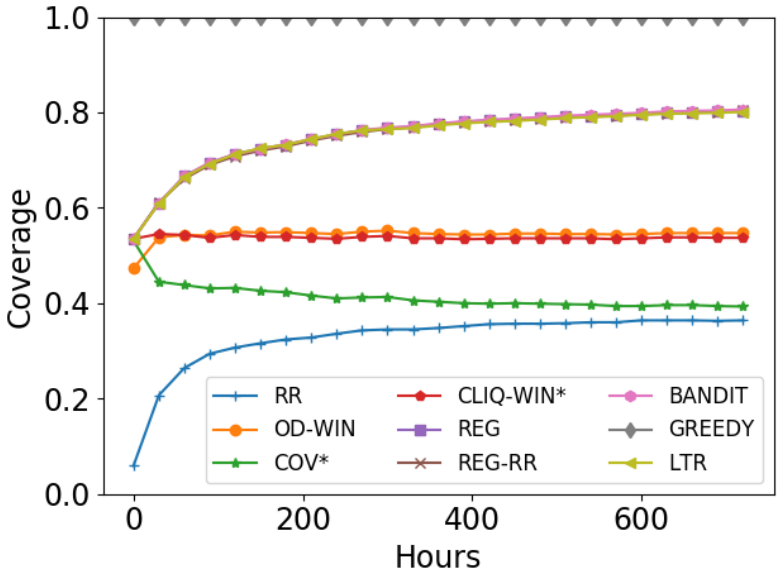



**Figure 14. Comparison of Coverage between Baselines and Proposed Methods for the Political News Domain**

## 4.4 Understanding Web Site Behavior Based On User Agent

Web sites have adopted a variety of adversarial techniques to prevent web crawlers from retrieving their content. While it is possible to simulate users behavior using a browser to crawl such sites, this approach is not scalable. Therefore, understanding existing adversarial techniques is important to design crawling strategies that can adapt to retrieve the content as efficiently as possible. Ideally, a web crawler should detect the nature of the adversarial policies and select the most cost-effective means to defeat them. In this work, we discuss the results of a large-scale study of web site behavior based on their responses to different user-agents. We issued over 9 million HTTP GET requests to 1.3 million unique web sites from DMOZ using 6 different user-agents (see Table 2) and the TOR network as an anonymous proxy. To reduce the risk that sites can identify our experiment and to reduce the chance the content changes in between requests, requests with different user-agents were sent from independent machines (with different IP addresses) and concurrently.

**Table 2. User-agent strings used in the experiments**

| Crawler Name | User-Agent String |
|---|---|
| ACHE | Ache |
| Bing | Mozilla/5.0 (compatible; bingbot/2.0; +http://www.bing.com/bingbot.htm) |
| Google | Mozilla/5.0 (compatible; Googlebot/2.1; +http://www.google.com/bot.html) |
| Browser | Mozilla/5.0 (X11; Linux x86 64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/42.0.2311.135 Safari/537.36 |
| Nutch | Nutch |
| Empty | |

The analysis of the responses uncovered many interesting facts and insights that are useful for designing adversarial crawling strategies at scale. For example, we observed that web sites do change their responses depending on user agents and IP addresses. Also, requests from less known crawlers have a higher chance of success. In contrast, when a TOR proxy is used, not only are most requests unsuccessful, but there is also a large number of exceptions. Another important finding is that response patterns vary for different topics – sensitive topics result in a larger number of 403 (forbidden) responses. These findings suggest that probing sites for these features can be an effective means to detect adversarial techniques. For more detailed results of the analysis, we refer to our paper [16].

## 4.5 ACHE Integration with Other Systems and Impact

ACHE has been released using the Apache open source license and can be downloaded from https://github.com/ViDA-NYU/ache. Its documentation is available at http://ache.readthedocs.io.

ACHE was integrated with the Domain Discovery Tool (DDT). It serves as the crawling engine that supports all crawling tasks available in DDT. ACHE also allows users to import a domain model built using the DDT and to start focused crawls or deep crawls (on sites specified by the user) using a simple and intuitive web.

ACHE is also integrated into systems developed by external research groups, such as the "myDIG web service" system developed by the Center on Knowledge Graphs from the Information Sciences Institute of the Southern University of California. We are also aware of start-ups using ACHE in their web crawling infrastructure in production. We were contacted by multiple parties, including start-ups from other countries interested in using the software. ACHE code repository and issue tracking system are hosted on the GitHub platform, where we regularly receive questions and feature requests. At the time of writing of this report, ACHE had reached 148 "stars" (number of times bookmarked as a favorite) and it keeps increasing, which indicates that the system is becoming more popular.

# 5 Conclusions

In this project, we pursued new research and developed new methods and tools that enable subject matter experts to effectively discover and track information on the Web that is relevant to a given task (or domain). We have designed and implemented: a new framework that facilitates domain discovery, organization and presentation, which enables users to seamlessly explore the content and create a computational model to recognize new content in the domain, and a general and extensible crawling infrastructure. Our research resulted in a novel solution to discover and gather domain-specific Web information at scale, addressing key challenges set forth by the DARPA Memex program. The results of this work have been disseminated as papers and open-source tools.

**Acknowledgments.** Many people have contributed to this project, including: Juliana Freire, Aline Bessa, Tuan-Ahn Hoang-Vu, Sonia Castelo, Yeuk Chan, Ari Juels, Clement Goubet, Shorya Gupta, Nanda Kalindindi, Yamuna Krishnamurthy, Qazi Munaf, Luis Gustavo Nonato, Aécio Santos, Monil Shah, Ritika Shandilya, Claudio Silva, Torsten Suel, Jorge Ono, Masayo Ota, Cesar Palomo, Rajat Pawar, Kien Pham, Deepti Verma, Ya Zhu, and staff members at Continuum Analytics.

# 6 References

[1] Krishnamurthy, Yamuna and Pham, Kien and Santos, Aecio and Freire, Juliana. “Interactive web content exploration for domain discovery”. *Interactive Data Exploration and Analytics (IDEA), KDD*, 2016.
[2] Gerard Salton and Christopher Buckley. “Term-weighting Approaches in Automatic Text Retrieval”. *Information Processing Management*, **24**, 1988, pp. 513–523.
[3] Ghahramani, Zoubin and Heller, Katherine A., “Bayesian sets”. *Advances in Neural Information Processing Systems (NIPS)*, 2005.
[4] Chakrabarti, Soumen and van den Berg, Martin and Dom, Byron. “Focused crawling: a new approach to topic-specific Web resource discovery”. *Proceedings of WWW*, 1999, pp. 1623–1640.
[5] Barbosa, Luciano and Freire, Juliana. “An adaptive crawler for locating hidden-Web entry points”, *Proceedings of the 16th international conference on World Wide Web*, 2007, pp. 441-450.
[6] Cortes, Corinna and Vapnik, Vladimir. “Support-vector networks”. *Machine Learning*, **20**, 1995, pp. 273–297.
[7] Bottou, Léon. “Large-Scale Machine Learning with Stochastic Gradient Descent”. *Proceedings of COMPSTAT*, 2010, pp. 177–186.
[8] Platt, John C. “Probabilistic Outputs for Support Vector Machines and Comparisons to Regularized Likelihood Methods”. *Advances in Large Margin Classifiers*, 1999, pp. 61–74.
[9] Ono, Jorge and Sikansi, Fabio and Correa, Debora and Paulovich, Fernando and Paiva, Afonso and Nonato, Luis Gustavo. “Concentric radviz: Visual exploration of multi-task classification”. *SIBGRAPI*, 2015, pp. 165–172.
[10] Anirban Dasgupta, Arpita Ghosh, Ravi Kumar, Christopher Olston, Sandeep Pandey, and Andrew Tomkins, “The discoverability of the web“, In *Proceedings of the 16th International Conference on World Wide Web*, 2007, pages 421–430.
[11] Ravi Kumar, Kevin Lang, Cameron Marlow, and Andrew Tomkins, “Efficient discovery of authoritative resources”, In *Proceedings of the 2008 IEEE 24th International Conference on Data Engineering*, 2008, pages 1495–1497.
[12] Peter Auer, Nicolò Cesa-Bianchi, and Paul Fischer, “Finite-time analysis of the multiarmed bandit problem”, *Machine Learning*, May 2002, 47(2-3):235–256.
[13] Karane Vieira, Luciano Barbosa, Altigran Soares da Silva, Juliana Freire, and Edleno Moura, “Finding seeds to bootstrap focused crawlers“, *World Wide Web*, 2016, 19(3):449–474.
[14] Luciano Barbosa, Srinivas Bangalore, Vivek Kumar, and Sridhar Rangarajan, “Crawling back and forth: Using back and out links to locate bilingual sites“, In *Proceedings of the 8th International Joint Conference on Natural Language Processing*, 2011.
[15] Kien Pham, Aécio Santos, and Juliana Freire, “Learning to Discover Domain-Specific Web Content” In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*. ACM, New York, NY, USA, 2018 432-440.
[16] Kien Pham, Aécio Santos, and Juliana Freire, “Understanding Website Behavior based on User Agent”, In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval*, ACM, New York, NY, USA, 2016, 1053-1056.
[17] L. Barbosa and J. Freire. An adaptive crawler for locating hidden web entry points. In WWW, pages 441–450, 2007.

[18] S. Chakrabarti. Focused web crawling. In Encyclopedia of Database Systems, pages 1147–1155. Springer, 2009.
[19] S. Chakrabarti, M. van den Berg, and B. Dom. Distributed hypertext resource discovery through examples. In VLDB, pages 375–386, 1999.
[20] G. T. de Assis, A. H. F. Laender, A. S. da Silva, and M. A. Goncalves. The impact of term selection in genre-aware focused crawling. In SAC, pages 1158–1163, 2008.
[21] M. Diligenti, F. Coetzee, S. Lawrence, C. L. Giles, and M. Gori. Focused Crawling Using Context Graphs. In VLDB, pages 527–534, 2000.
[22] F. Menczer, G. Pant, and P. Srinivasan. Topical web crawlers: Evaluating adaptive algorithms. ACM Trans. Internet Techn., 4(4):378–419, 2004.
[23] S. Sizov, M. Theobald, S. Siersdorfer, G. Weikum, J. Graupmann, M. Biwer, and P. Zimmer. The bingo! system for information portal generation and expert web search. In CIDR, 2003.
[24] M. L. Vidal, A. S. da Silva, E. S. de Moura, and J. M. B. Cavalcanti. Structure-driven crawler generation by example. In ACM SIGIR, pages 292–299, 2006.

# 7 Appendix

### Open Source Systems

The ACHE Crawler. https://github.com/ViDA-NYU/ache

Domain Discovery Tool. https://github.com/ViDA-NYU/domain_discovery_tool

### Publications (that describe part of the work carried out in this project)


- Interactive Multi-Scale RadViz with Co-Clustering to Explore Text Collections. Sonia Castelo, Yamuna Krishnamurthy, Jorge Piazentin Ono, Claudio T. Silva, and Juliana Freire. *Submitted for publication, 2018.*
- Learning to Discover Domain-Specific Web Content. Kien Pham, Aecio Santos, and Juliana Freire. *Submitted for publication, 2018.*
- Real-time understanding of humanitarian crises via targeted information retrieval. Kien Pham, Prasanna Sattigeri, Amit Dhurandhar, Arpith Jacob, Maja Vukovic, Patrice Chataigner, Juliana Freire, Aleksandra Mojsilovic, and Kush Varshney. IBM Journal of Research and Development, 61(6), pp 7:1–7:12, 2017.
- Finding seeds to bootstrap focused crawlers. Karane Vieira, Luciano Barbosa, Altigran Soares da Silva, Juliana Freire, Edleno Moura. World Wide Web 19(3): 449-474, 2016.
- Interactive Exploration for Domain Discovery on the Web. Y. Krishnamurthy, K. Pham, A. Santos, and J. Freire. In ACM KDD Workshop on Interactive Data Exploration and Analytics (IDEA), pp. 64-71, 2016.
- Understanding Website Behavior based on User Agent. Kien Pham, Aécio S. R. Santos, and Juliana Freire. In ACM SIGIR, pp. 1053-1056, 2016.
- ACHE: Documentation. http://ache.readthedocs.io
- DDT: Documentation. http://domain-discovery-tool.readthedocs.io

**Other Publications (that acknowledge the grant)**

- ARIES: Enabling Visual Exploration and Organization of Art Image Collections. Lhaylla Crissaff, Louisa Ruby, Samantha Deutch, Luke DuBois, Jean-Daniel Fekete, Juliana Freire, and Claudio T. Silva. IEEE Computer Graphics & Applications (CG&A), 38(1): 91-108, 2018.
- Interactive Visual Exploration of Spatio-Temporal Urban Data Sets using Urbane. Harish Doraiswamy, Eleni Tzirita Zacharatou, Fabio Miranda, Marcos Lage, Anastasia Ailamaki, Claudio T. Silva, and Juliana Freire. In ACM SIGMOD, 2018. Best demo award.
- Querying and Exploring Polygamous Relationships in Urban Spatio-Temporal Data Sets. Yeuk-Yin Chan, F. Chirigati, H. Doraiswamy, C. Silva, and J Freire. In ACM SIGMOD, pp. 1643-1646, 2017. Best demo honorable mention.
- GPU Rasterization for Real-Time Spatial Aggregation over Arbitrary Polygons. Eleni Tzirita Zacharatou, Harish Doraiswamy, Anastasia Ailamaki, Claudio Silva, and Juliana Freire. In PVLDB vol. 11(2), pp. 352-365, 2017.
- Time Lattice: A Data Structure for the Interactive Visual Analysis of Large Time Series. Fabio Miranda, Marcos Lage, Harish Doraiswamy, Charlie Mydlarz, Justin Salamon, Yitzchak Lockerman, Juliana Freire, and Claudio Silva. In Proceedings of Eurographics Conference on Visualization (EuroVis), CG&A, 37(3), pp. 23-35, 2018.
- Learning to Discover Domain-Specific Web Content. Kien Pham, Aecio Santos, and Juliana Freire. In ACM International Conference on Web Search and Data Mining (WSDM), pp. 432–440, 2018.
- Exploring What not to Clean in Urban Data: A Study Using New York City Taxi Trips. J. Freire, A. Bessa, F. Chirigati, H. Vo, and K. Zhao. IEEE Data Engineering Bulletin, 39(2):63-77, 2016.