Sifter

Sifter is a search application for digital forensics investigators. It indexes text from digital media, including file slack and unallocated space, and allows the investigator to query the index with keywords for responsive documents. Additionally, Sifter groups related documents together into clusters and graphically displays the clusters to the investigator. In this way, the investigator can explore textual evidence without predefined search terms, and find related documents that may not be responsive to a particular search term.

Sifter requires that a Java Runtime Environment (JRE) be installed, preferably 64-bit. A JRE for your platform may be downloaded for free from Oracle's website. The Java browser plugin, which has been the source of many security vulnerabilities, is ***not*** required for running Sifter.

The command-line examples given below are for Windows, but Sifter is entirely cross-platform.

Indexing

To index an evidence file, open a command-line terminal and change directory to be in the Sifter program folder. Then issue the following command:

```
C:\Sifter>.\bin\fsrip.exe --unallocated=block dumpfiles Eviden
ce.E01 |
.\index_evidence.bat IndexDirectory stoplists\stoplist_winXP.t
xt
```

This command runs the fsrip utility, which knows how to read files out of digital media, and pipes its output to the Sifter indexing utility. *Evidence.E01* is the evidence file and *IndexDirectory* is a path to a new top-level directory where you'd like the index to be created. One of the stoplist files from the Sifter stoplists directory should also be specified. Stoplists are provided for different operating systems, and should correspond to the operating system on the evidence file. Indexing will produce a great deal of console output, which is useful for debugging if an error occurs. At the end, you will see output like this:

```
Successful finish
FilesRead: 290792
BytesRead: 1923952968
FileBytesRead: 1735457181
Duration: 578 seconds
```

Sifter attempts to extract the text of each file with the Apache Tika library. You will probably see error messages during indexing, related to exceptions from the text extraction process. This is normal. If Tika fails, then low-level text scraping (with UTF-8, followed by UTF-16) is attempted. Sifter indexes file slack and unallocated blocks as separate documents. Files that do not have any content (e.g., null clusters) are not indexed.

By using a pipe to communicate, the files from evidence do not need to be written out temporarily in order to be indexed. Sifter's indexing operation is able to buffer the input and index most files in RAM (very large files will be written to disk as temporary files to avoid exhausting memory). When a file is read in, its content is handed to a separate worker thread that extracts the text from the file using Tika and then indexes the text and accompanying filesystem metadata using Apache Lucene. The total amount of memory used during indexing is largely determined by multiplying the large file threshold size (MB) by the number of threads in the pool.

For best performance, it is recommended that the evidence file and the index directory be located on separate storage devices.

Clustering

After indexing has completed, clustering can be initiated. Clustering uses the index as its only source of input, so this can happen on a machine which does not have the evidence file. To generate the cluster, run this command from the command-line:

```
C:\Sifter>.\make_som.bat Index_Directory
```

This reads data from the index and organizes each document into a feature vector, where the top terms in a document are represented as a binary vector. Each binary feature vector is then fed to a Self-Organizing Map (SOM, also known as a Kohonen Map) algorithm, which gradually places related feature vector into nearby cells in a two-dimensional grid.

Clustering also generates output to give you an idea of how far along it is in processing. At the end, it will show output like this:

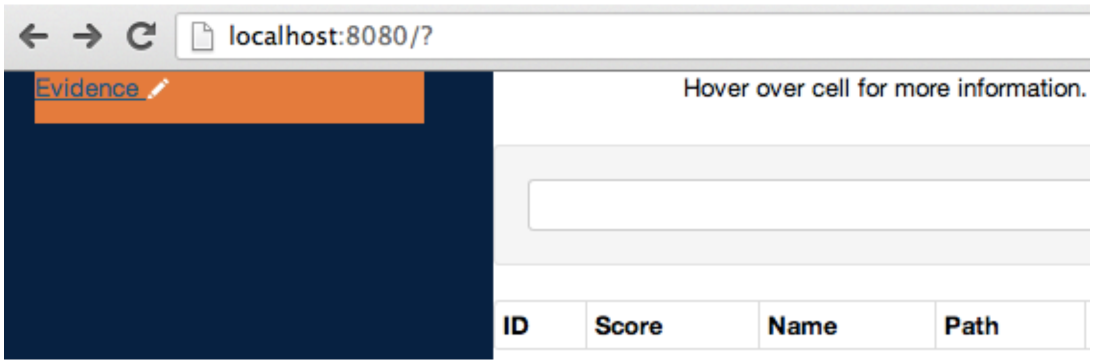
```
processed 108216 docs in this iteration, 75608 had closest cel
ls
Finished iteration 3
Assigning documents to clusters
Rescaling SOM vectors
Assigning top terms to each cell
Calculating greatest neighbor term difference
Assigning cells to regions
Writing final output
Number of docs written: 108216
Number of outlier docs: 32608
Total term dimensions: 3000
Max terms per doc: 2146
Avg terms per doc: 19.470624801608295
Duration: 186 seconds
```

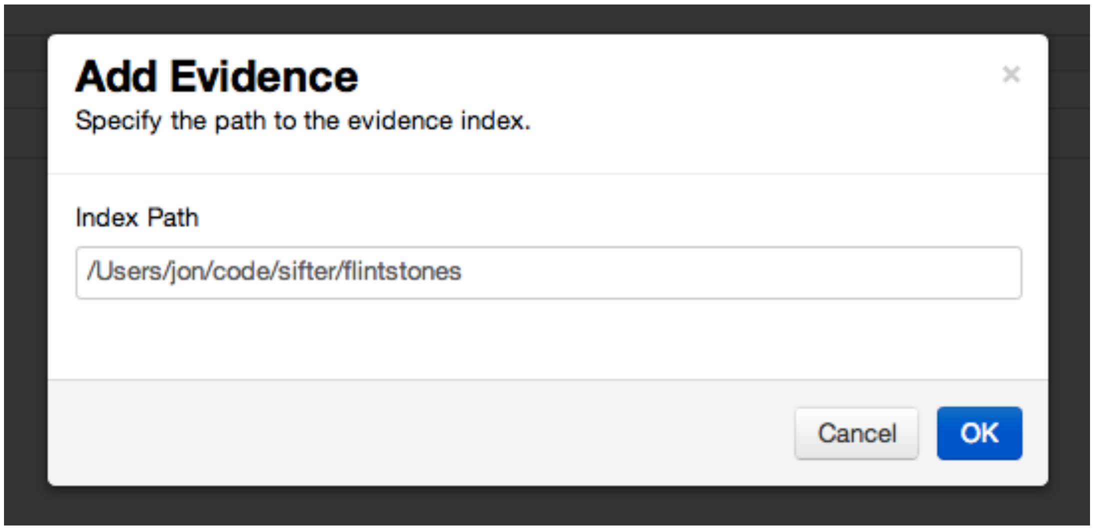
Review

After clustering has completed, the index is ready to be queried and reviewed in a web browser. Either double-click "Start_webserver.bat" or run it from the command-line. Then open your web browser and go to http://localhost:8080 in the URL bar. This will bring up the Sifter user interface. Sifter uses an embedded Jetty web server and does not need separate configuration.

Opening Evidence

Click on the "Evidence" link in the upper-lefthand corner and enter in the full path to the index directory into the "Add Evidence" pop-up dialog.

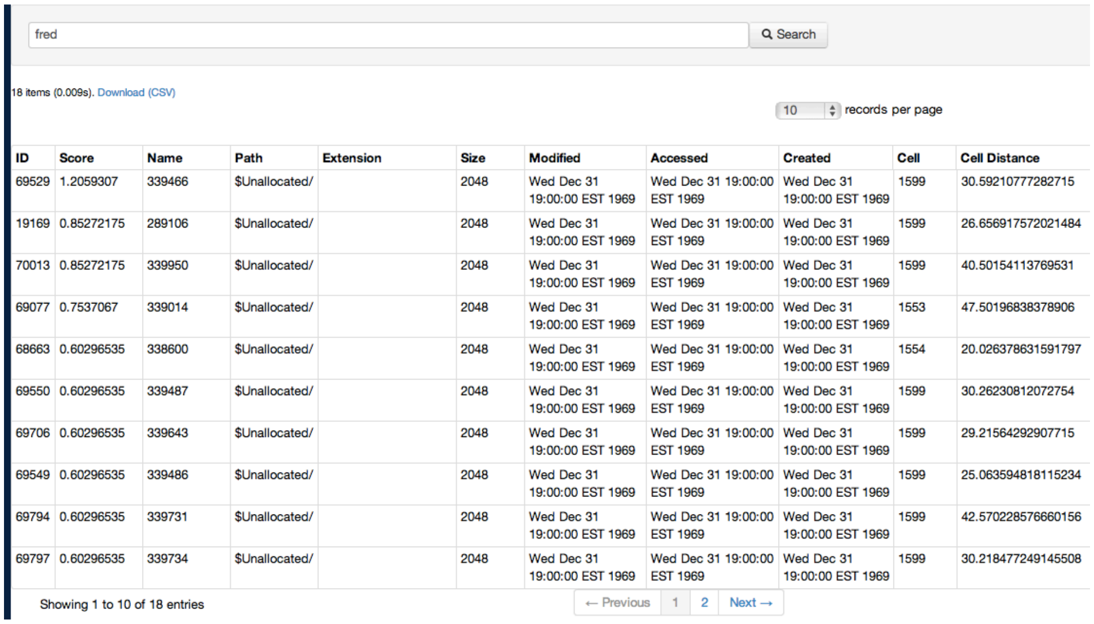



Add Evidence
Specify the path to the evidence index.
Index Path
/Users/jon/code/sifter/flintstones
Cancel
OK

Click "OK" and the Self-Organizing Map graphic will be displayed, as well as a search box below it. Mouse-over the colored cells in the SOM and click on them for more detail about the documents related to each cell. Enter search terms in the search box to retrieve a table of documents responsive to the query.

Search Queries

Search queries may be entered in the text box below the SOM graphic. This is a "Google-style" index search, where the results are returned as individual documents, sorted by a relevancy score. The responsive documents are shown in a table below the search box, with columns for different metadata fields.



18 items (0.009s). Download (CSV)

10 records per page

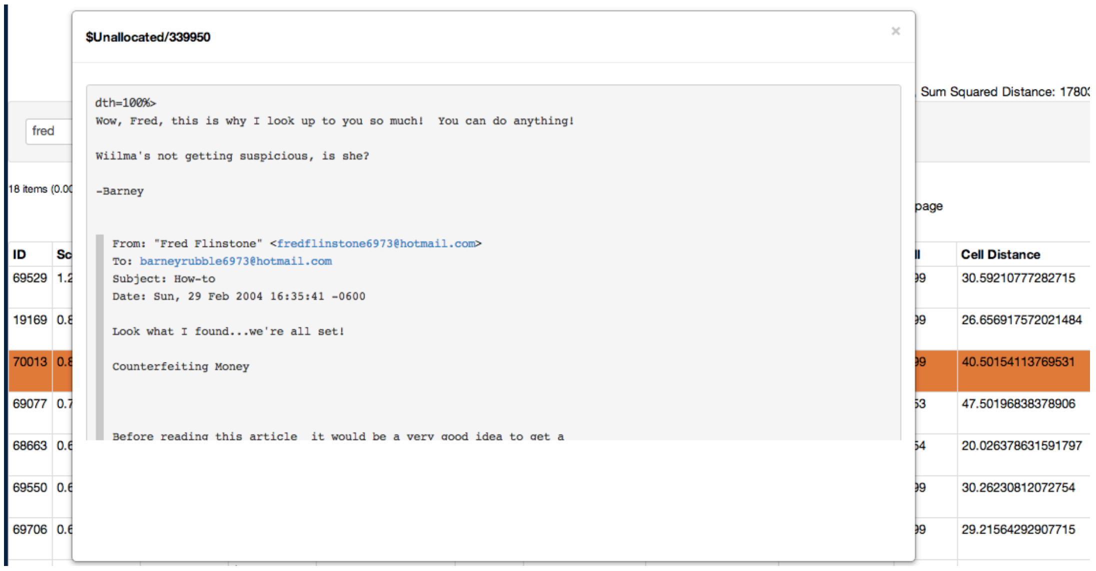
ID	Score	Name	Path	Extension	Size	Modified	Accessed	Created	Cell	Cell Distance
69529	1.2059307	339466	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1599	30.59210777282715
19169	0.85272175	289106	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1599	26.656917572021484
70013	0.85272175	339950	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1599	40.50154113769531
69077	0.7537067	339014	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1553	47.50196838378906
68663	0.60296535	338600	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1554	20.026378631591797
69550	0.60296535	339487	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1599	30.26230812072754
69706	0.60296535	339643	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1599	29.21564292907715
69549	0.60296535	339486	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1599	25.063594818115234
69794	0.60296535	339731	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1599	42.570228576660156
69797	0.60296535	339734	$Unallocated/		2048	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	Wed Dec 31 19:00:00 EST 1969	1599	30.218477249145508

Showing 1 to 10 of 18 entries

The total number of responsive documents is shown above the table, followed by the time the query took to execute (often less than 1 second), and a link to download the result set as a CSV file.

Search queries use [Lucene's built-in syntax]() and can be quite advanced, with support for Boolean logic across different fields, ranges, and term proximity in the document body. All documents can be listed simply by clicking the "Search" button with an empty query.

The extracted text from documents can be viewed in a pop-up dialog by clicking on the row.



$Unallocated/339950

```
dth=100%>
Wow, Fred, this is why I look up to you so much!  You can do anything!

Wiilma's not getting suspicious, is she?

-Barney


  From: "Fred Flinstone" <fredflinstone6973@hotmail.com>
  To: barneyrubble6973@hotmail.com
  Subject: How-to
  Date: Sun, 29 Feb 2004 16:35:41 -0600

  Look what I found...we're all set!

  Counterfeiting Money



  Before reading this article  it would be a very good idea to get a
```

Sum Squared Distance: 1780

fred

18 items (0.0

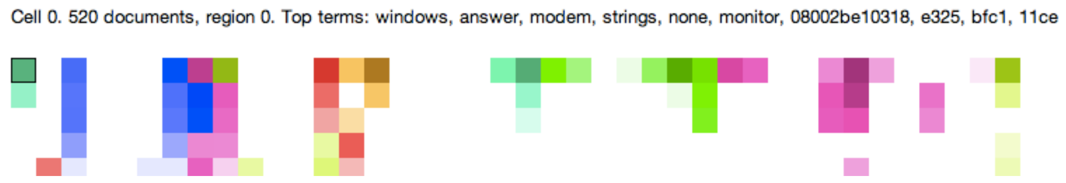
page

ID	Sc	ll	Cell Distance
69529	1.2	99	30.59210777282715
19169	0.8	99	26.656917572021484
70013	0.8	99	40.50154113769531
69077	0.7	53	47.50196838378906
68663	0.6	54	20.026378631591797
69550	0.6	99	30.26230812072754
69706	0.6	99	29.21564292907715

SOM Navigation

The self-organizing map graphic represents similar documents in a 2-dimensional grid. Similar documents are placed into clusters, which each cluster represented by a cell in the grid. Neighboring cells tend to be similar to each other as well.

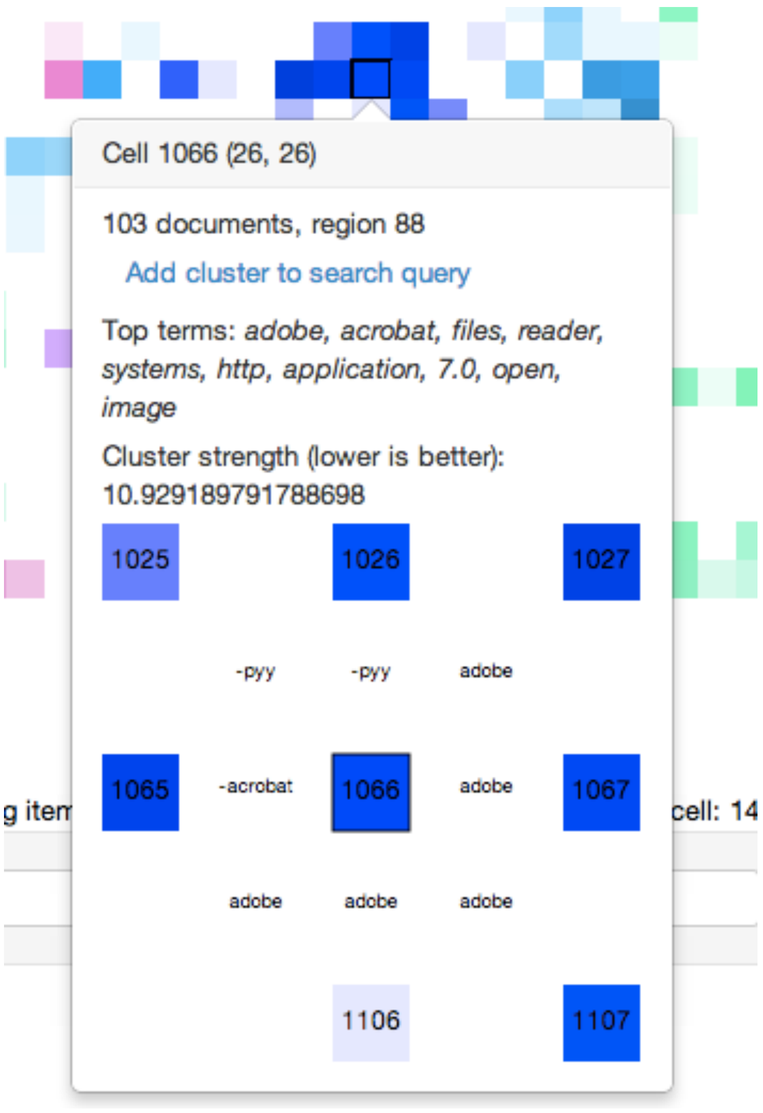
Hovering the mouse pointer over a cell will display basic information about the cell above the SOM graphic. This includes the cell identifier, the number of documents in the cluster and the most frequently occurring words (in order) in the cluster's documents.



Cell 0. 520 documents, region 0. Top terms: windows, answer, modem, strings, none, monitor, 08002be10318, e325, bfc1, 11ce

Cells which share the same most frequently occurring word are grouped into the same region. Cells in the same region share the same hue of color, with region colors assigned randomly like colors on a map. Darker cells contain more documents than lighter cells, and brightly colored cells indicate that the documents in the cell are more strongly related to each other than cells where the color appears dim or grayish.

Clicking on a cell displays more information about the cell and its relationship to its neighboring cells. Clicking on "Add cluster to search query" will show all the documents in the cell cluster to the search results table. The exploded grid view shows the biggest term difference between the selected cell and its neighbors. In the example below, we can infer that the cell has more occurrences of the word "adobe" than its neighbors to its right and bottom, and fewer occurrences of "pyy" and "acrobat" than its neighboring cells 1025 & 1026 and 1065, respectively.



Cell 1066 (26, 26)

103 documents, region 88

Add cluster to search query

Top terms: *adobe, acrobat, files, reader, systems, http, application, 7.0, open, image*

Cluster strength (lower is better): 10.929189791788698

The cell information can be hidden by re-clicking on the cell. Many popovers can be open at a time.

Algorithm Information

Sifter implements the Scalable Self-Organizing Map algorithm described by Roussinov (need citation). Rather than update all weights in a cell vector with values from a document term vector, the Scalable SOM algorithm uses binary term features and only updates cell weights coincident with a given document's features (using some moderately complicated algebra). This effectively means that Sifter can handle thousands of features whereas a typical SOM (or other typical machine learning algorithms) can only handle a few hundred at most. This helps when attempting to confront the diversity of data on digital media.

The amount of memory used by Sifter during the clustering process has a lower bound of `8 bytes * number of features * SOM height * SOM width`. To avoid storing all document term vectors in RAM, Sifter writes them out to a binary file and streams through it, in order, for each iteration. Because it uses an efficient serialization form and makes use of buffering, using the file does not represent a significant performance hit.

The Scalable SOM algorithm is an iterative algorithm, and not inherently data parallel. This limits the benefit of multithreading/multiprocessing. However, Sifter will parallelize some of its operations in a fine-grained manner. The larger the SOM dimensions, the more benefit will be seen from multithreading.

The most frequent terms below the doc_freq_threshold_high threshold are selected as input features.

On a typical disk image, there will be many files which do not contain any of the selected terms. These are considered outliers. Outliers can be displayed in the table view with a search query of `cell:"-1"`. Improving outlier analysis remains an open problem.

Configuration

Sifter uses a number of parameters stored in an XML file to control its operation.

This file is "sifter_props.xml". You may need to edit the parameters first in order to run Sifter (in particular, you may need to change the temporary directory).

The properties are:

- thread_pool_size

  The number of background threads to use for indexing and clustering. Ideally it should be 5-8, depending on the number of CPU cores on your workstation.

- random_seed

  An integer that's used to seed the random number generator used by the SOM algorithm. This is specified to allow for deterministic results and does not need to be changed.

- indexing_buffer_size

  The size, in megabytes, of the indexing buffer used by Apache Lucene. Best performance by setting this to between 64-128.

- large_file_threshold

  The size, in megabytes, of each worker thread's memory buffer for storing file contents. Files larger than this will be exported to the temporary directory for indexing. Note that the more workers in the threadpool, the more RAM will be used.

- temp_dir

  The directory where large files will be exported temporarily for indexing. The directory must already exist or else Sifter will generate an error (this will be fixed).

- doc_freq_threshold_high

  A number from 0-1, this number is the document frequency cutoff to use for choosing term features. A term that occurs in more than this number of documents relative to the number of documents in the index will not be used as a feature. The default is 0.66, i.e., a term that occurs in more than two-thirds of documents will not be used for the SOM clustering.

- doc_freq_threshold_low

  A number from 0-1, this number is the minimum document frequency threshold in order for a term to be considered for SOM clustering. The default is 0.0001.

- min_som_term_length

  The minimum length of a term to use for SOM clustering. Words/terms shorter than this length will not be considered for SOM clustering. The default is 3.

- max_vector_features

  The maximum number of features to use for SOM clustering. A few hundred should be used at least, and several thousand can be used.

- som_width

  The number of cell columns in the SOM.

- som_height

  The number of cell rows in the SOM.

- max_neighbor_radius

  The maximum distance used for updating cells in the SOM on the first iteration. Assuming a square SOM, a good choice is to use one-quarter of the SOM's dimension.

- min_neighbor_radius

  The minimum distance used for updating cells in the SOM on the last iteration. A good choice is 1.

- max_alpha

  A number from 0-1, the amount to update a SOM cell with each assigned document on the first SOM iteration. A good choice is .0001.

- min_alpha

  A number from 0-1, the amount to update a SOM cell with each assigned document on the last SOM iteration. A good choice is .00001.

- num_som_iterations

  The number of SOM iterations to perform before outputting the final cluster. Should be at least 2.

- num_top_cell_terms

  The number of top cell terms to display in the web application. The default is 20.