Master Project Report

Author
Raül Rabionet Janssen

Development of a tool for the identification of counterfeiters in the internet

Project Supervisor
Dr. Martin Rajman
# TABLE OF CONTENTS

1 DESCRIPTION OF THE PROBLEM ........................................................................... 3
  1.1 Motivation ........................................................................................................ 3
  1.2 How is Counterfeiting Being Fought Against ......................................................... 3
  1.3 Technological State of the Art: The Internet Patrolling Project ................................. 4

2 DESIGN OF A SOLUTION .................................................................................... 6
   2.1 Architecture of the Prototype ........................................................................ 6
      2.1.1 Module 1: Document Gathering and Entity Extraction ............................. 6
      2.1.2 Module 2: Link-Based Analysis ................................................................ 6
      2.1.3 Module 3: Results output and Multi-document Information Extraction .............. 7
   2.2 Integration of the FAST Platform ....................................................................... 7

3 IMPLEMENTATION OF A PRELIMINARY SOLUTION ........................................ 9
   3.1 Goals of the Demonstrator ............................................................................. 9
   3.2 Architecture of the Demonstrator ................................................................... 9
      3.2.1 Module 1: Document Retrieval and Entity Extraction .............................. 9
      3.2.2 Module 2: Document Clustering ............................................................... 10
      3.2.3 Module 3: Case Frames and Results output .............................................. 10

4 IMPLEMENTATION OF MODULE 1 – DOCUMENT GATHERING AND ENTITY EXTRACTION .......................... 10
   4.1 Data Flow ........................................................................................................ 11
   4.2 Generation of a Seed List ............................................................................... 12
   4.3 Web Crawling and Document Retrieval ......................................................... 13
   4.4 FAST processing ............................................................................................ 16
   4.5 WHOIS Querying ........................................................................................... 16
   4.6 Entity Extraction and Normalization ............................................................... 18
   4.7 Document and Entity Output ......................................................................... 20
   4.8 Page Scoring and Entity Selection .................................................................. 21
   4.9 Second Stage Crawl ....................................................................................... 22

5 IMPLEMENTATION OF MODULE 2 – DOCUMENT CLUSTERING .......................... 23
   5.1 Generation of the DocVectors ......................................................................... 24
   5.2 Reloading Documents ..................................................................................... 25
   5.3 Document Clustering ...................................................................................... 26
   5.4 Cluster Labeling ............................................................................................. 26

6 MODULE 3 - RESULTS OUTPUT ........................................................................ 27

7 TESTS AND EVALUATION ............................................................................... 28

8 FUTURE WORK ON IMPLEMENTATION .......................................................... 29
1 Description of the Problem

1.1 Motivation

The International Anticounterfeiting Coalition (a non-profit organization devoted to fighting counterfeit globally) estimates that counterfeits caused $600 billion losses to companies worldwide in 2006. According to Swissinfo, the damage to the Swiss economy on 2006 amounted for $1.6 billion – 650 Million of which correspond to the watch industry. The OECD estimated on a report from 1998 that 40 Million watches were being produced by counterfeiters around the globe every year, to the 25 Million being legitimately produced at the time.

BASCAP, an initiative of the International Chamber of Commerce to fight Counterfeiting, publishes yearly studies which summarize the reported counterfeiting activity. According to those, the sector of counterfeited pharmaceutical products escalates to the third position in the ranking of most counterfeited products (second only to Software and Clothing) – the impact of these counterfeits on the global economy is estimated around $40 billion.

All the institutions involved in the control and prosecution of counterfeiting agree that the number of cases and their economic relevance has rocketed up in the last years, mainly due to the use of the internet to distribute the counterfeited products. According to the World Health Organization, 50% of the drugs and medical products currently being sold through the Internet are counterfeited, posing serious hazards not only to the business of pharmaceutical companies, but also to the health of the consumers.

Due to the anonymity and globality provided by Internet, most counterfeiters are becoming openly blatant on their activity. Using a standard search engine on the internet and keywords as clear as "Replica" or "Fake", we have spotted hundreds of different websites, offering full portfolios of counterfeited luxury products.

There are also available in the internet pages that compare the quality of different counterfeiters, working as a marketing and information platform for some of them, such as http://www.replicacentre.com

1.2 How is counterfeiting being fought against

Most Business Coalitions and global organizations on commerce and trade, such as the WTO or the World Customs Organization, as well as the EU and European governments have been focusing in the need that growing industrial powers such as the countries emerging in south-eastern Asia respect intellectual and industrial property rights as they are defined and defended by better established economies. Thanks to these efforts, countries like China are updating their Intellectual Property Legislation.

Nevertheless, a lot of counterfeiting activity takes place even inside the borders of the United States or the European Union. Existence of such laws is not the only obstacle preventing the prosecution of the counterfeiters. Enforceability of these laws, as well as the ability to properly identify the counterfeiting agents and provide evidence of their activity, are much harder problems.

According to BASCAP, 760 incidents of intellectual property theft where caught during the first half of 2006, for a value of $700 Million in seizures and losses and involving 1.630 individuals. This is still only the tip of the iceberg.

Most consulting companies are providing solutions to the counterfeiting problem, but these solutions mostly consist in more elaborate authentication methods, to make it more difficult to produce exact replicas, allowing well-informed consumers to spot fake products easily. The
following list of IACC members includes links to some of these consulting companies: http://www.iacc.org/anticounterfeiting/member.php.

These solutions might be becoming less efficient as the replicas' quality increases. Although this is not the case for pharmaceutical products, some replicas of luxury and clothing items are achieving similar levels of quality as their originals, thus profiting of the companies' investment in design and marketing while being able to publicly admit to be counterfeits.

Based on some additional market analysis done during the recent months, it has become clear that the largest luxury groups who would targeted by the future developed application are mainly using well known internet search platforms (i.e google) to search manually for the information and counterfeit sites. These eCommerce sites are well publicized and accessible through a public internet search engine. These manual searched are mainly done for monitoring purposes, but the actual investigation are outsourced to legal firms which use investigation methods on the field to try to match an internet channel with a network of counterfeiters, manufacturers and resellers. This is mainly done on a case by case basis. No comprehensive search and data extraction is done automatically with an application as suggested in the project proposal.

We have also seen some specific tools for monitoring blogs or auction sites, but again, these solution are rather a point solution to monitor a specific topic or activity over a specific site, but not with a view of create a comprehensive data warehouse related to the commerce of luxury counterfeit products to allow reporting, monitoring and legal case support which is the aim of our project.

Today we see a lack of information and reporting on the status of the counterfeit products sold over the internet, the average price and characteristics of that market.

1.3 Technological State of the Art: The Internet Patrolling Project

As exposed above, most current techniques oriented to fight against counterfeiting rely on developing authentication methods that can be applied to the products on the design and production steps. This does not prevent counterfeiting from happening, only hindering counterfeiters’ efforts to produce exact copies.

In order to effectively fight against the companies and persons behind the grey market, we need to lift the veil of anonymity provided by the internet. The Internet Patrolling project was born in EPFL with the intention of providing a tool that would work on that direction. The tool we are presenting here has been developed as first prototype of the Internet Patrolling Project and implements a basic version of what is expected to be the final result of the project, according to the following structure:

1. Step 1: Document gathering and entity extraction:
   In this first module, the system tries to automatically identify and process websites working on counterfeited goods, as well as other pages related to these ones, in order to obtain evidence of the illegal activity and retrieve data on products being counterfeited, contact and payment methods that can lead to the intervening parties and any other such information associated to the entities identified, which can be obtained from other websites using previously discovered data like IP addresses o telephone numbers as search words.
   In order to do this we apply techniques of web mining and information retrieval, applying where possible the pre-existent tools developed within the FAST platform.
The NEMIS European Network, in which the Artificial Intelligence Laboratory of EPFL took part, put a lot of interest in developing a series of Web Mining methods that might be useful for the present project.

The techniques to be used to capture the relevant data are those related with Text Mining, Information Extraction and the text search through keywords and prediction rules. The IP project is collaborating with Dr. Ronen Feldman, a reputed specialist in Text Mining and Information Extraction from unstructured texts\(^1\), who has proven these techniques to be useful for the retrieval of factual data in other fields (such as biomedical documentation).

2. Step 2: Link-based analysis
The "most relevant" pages are identified, using page ranking and scoring techniques. After this is done, these pages will be represented in a graph, and linked to each other based on the sharing of relevant pieces of information. We plan to connect the information obtained to form clusters that can help us identify useful information and put together all data referring to a single individual.

This link-based analysis has been successfully applied in previous projects of distributed information extraction, such as the study by the Los Alamos National Laboratory on detecting and interrelating terrorist cells\(^2\) ("Advanced Knowledge Integration in Assessing Terrorist Threats"). Studies on the "Small World Theory\(^3\)" and other theories on network information extraction have also developed theoretical models for information gathering on the web. The abovementioned NEMIS project also put some effort on defining Web clustering algorithms based on different criteria.

3. Step 3: Multi-document Information Extraction
A set of frames of necessary and relevant information have been defined in cooperation with our industrial partners. This include all information useful to identify the counterfeiting agents and gather any further evidence or data useful to start legal procedures against the identified persons.

On this step of the process, the information needed to fill in such frame will be extracted from the processed websites. To do this we might be implementing different types of Ontology-Driven Information Extraction Algorithms and other Data Mining Algorithms.

These techniques have been successfully used for the extraction of specific information from distributed databases\(^4\), and have been extended to be combined with Information Extraction Rules and other Web Mining techniques to extend domain ontologies by extracting information directly from the web\(^5\).

Previous projects carried out at EPFL, such as the EXTRACT project, also worked on the use of knowledge models for automated information extraction out of non-structured texts. We will be applying those models in this module of the software.

A few commercial tools where located that offer similar features as what the IP project is pursuing: gathering enough information to take legal actions against the individuals behind the counterfeiting activities. Since these are developed by private consulting companies, there is little information on their actual performance or the underlying techniques.

As stated in the previous section, some of the main luxury companies affected by this counterfeiting phenomenon were contacted, and none of them are using any tool specifically developed to take care of this raising problem, often performing manually small surveillance tasks on the offending sites.
2  Design of a Solution

2.1  Architecture of the prototype

What follows is the general specification of the modules in within which the prototype is to be divided. This work of specification aims at integrating all the techniques described in the previous section. The implementation work that has been carried out this far does not cover all the specifications detailed hereunder; the next section in this document provides a detailed description of a preliminary implementation of the prototype, whereas some of these features have been simplified. Nevertheless, it is interesting to see what the final version of the prototype is intended to be.

2.1.1  Module 1: Document Gathering and Entity Extraction

In the first module of the process, we want to:

- Identify pages that are relevant to our investigation
- Download and process them, extracting from them the Entities that will be necessary in the following modules. Entities are relevant pieces of information, such as names, phone numbers or DNS names.

The identification of relevant pages is done using a seed generator and the crawler integrated in the FAST platform; the seed generator queries a standard searcher using a list of search words that are central to the investigation, and produces an initial list of URLs that are related to it. These URLs are used as starting points for a customizable crawl that will provide the documents from which the Entities will be extracted.

During the test stages of the prototype, we will be using the same search words to identify relevant pages. This search words are a combination of words that identify a brand and words that are related to counterfeiting. Once the prototype is running, we will have a means to test different word combination to provide a list of suggested keywords, such that the seed generator will use some of those keywords, together with the brand names, to generate the seed list of URLs.

The relevant pages are then downloaded and stored. They are processed using the FAST platform, in order to:

- Extract all relevant Entities.
- Complete the information with searches on other information bases.
- Weight the pages, according to their interest and the relevance of their Entities.
- Filter out irrelevant pages.

Using the page ranking and frequency information, the most significant Entities will be used to generate new seed lists — these lists will be fed to the crawler anew to download other pages, related not directly to the initial search but to the Entities that we are interested in. These pages will also be processed using the FAST platform, and added to the document collection.

2.1.2  Module 2: Link-Based Analysis

The output of module 1 is a collection of tagged documents, from which we are especially interested in the Entities they contain. These could be represented in a graph, within which the documents are nodes connected by edges representing shared Entities, such edges being weighted according to the quality of the Entities they represent.

We can assume that a document will be related to a large number of other documents, and maybe several times to the same document. We want to identify those relationships that are not purely random. The number of occurrences of a relationship could be relevant, and will be used
to weight the edges of the graph. Nevertheless, some of the information we want to use is not expected to appear repeatedly, and we should not discard single appearances, for in some cases a single occurrence of a useful Entity should be deemed sufficient (such as the phone number of a person appearing only once on, for example, his home page).

We will try to identify clusters of information, where a number of documents is tightly connected by appearing consistently together in different websites, or by forming cycles. These clusters of documents will be then ranked according to their size and to the scores of the affected pages, in order to select the most promising clusters to proceed to the next step.

2.1.3 Module 3: Results output and Multi-document Information Extraction

Once a cluster has been identified as a subset of documents tightly related among themselves, and relevant enough according to our ranking criteria, we will proceed to extract information from it.

The final goal of the prototype is to provide a useful output format, such as a frame of useful information that a potential client would be interested to obtain on each of the counterfeiting agents that affect their business. We will consider that a cluster of Entities corresponds to the activities of a single agent or a group of related agents, and therefore will try to obtain one frame out of each selected cluster. Multidocument Information Extraction techniques will be used to select the relevant information to be included in the frame.

2.2 Integration of the FAST Platform

The FAST Enterprise Search Platform is a complete document processing and searching tool developed and owned by the Norwegian company FAST

FAST ESP is a tool oriented at giving companies and organizations an easy way to access, interact with and exploit their information bases. It enables rich search management functionality and tuning capabilities, and makes it possible to detect the context and the intent of the query, search for terms and phrases, and return requested entities that appear in the context of the matching text.

FAST ESP provides secure access to available information and deliver this information in a secure way, and can access a wide variety of data sources: rich media data, formally structured data held in databases, semi-structured data, and unstructured information held in Web pages, e-mail, blogs, documents, presentations, and the like. It supports 80 languages and about 400 file formats.

It also grants built-in scalability and high performance, scaling linearly, independently, and simultaneously in three dimensions: data volume, queries per second, and freshness of data.

FAST is playing an important role in the development of the IP Project, being the main platform on which the project is being developed. The preliminary implementation that will be described hereunder is strongly based on FAST, so it is interesting to have a look at this platform.
FAST ESP includes, among other features, the following that will be most useful for us:

- A customizable and efficient web crawler, that can be automatically configured using .xml.
- A document processing and indexing tool that pushes each downloaded document into a pipeline of customizable processing stages before introducing it into it search index.
- Some well-tested Information Extraction modules that we can directly use or adapt when necessary, in order to extract the relevant Entities from each processed document. These modules include very advanced linguistics, especially in English, and efficient Regular Expression detectors.
- The possibility to search through the indexed documents using navigation tools such as dynamic drill down.

Since a large part of the work in this first step of the prototype consists in crawling web pages, storing them and identifying the relevant Entities, it seems natural to use an enterprise tool that is specifically developed to perform these tasks.

Therefore, a part of the work in this Master Project has consisted in the familiarization with the FAST platform, installing it, learning how to use it and how to develop and debug code specifically intended to work with it.

3 Implementation of a preliminary solution

3.1 Goals of the demonstrator
Following the specification of the prototype detailed in Deliverable 1.2, we set the following goals for the first implementation of the demonstrator:

- Having a working tool that implements the whole targeted process, even if it temporarily uses simpler solutions for some steps. Having such a full demonstrator will indeed allow for the subsequent improvements to be focused on specific steps, while being tested in a complete setting, thus providing a better prediction of what the final results will be, and how each modification will impact them;
- Running a full test on a controlled data set, in order to evaluate the quality of the results that can be obtained with the current implementation;
- Identifying the strengths and weaknesses of the current implementation, and guiding the improvement work.

The demonstrator is developed within the FAST platform, with only some additional processing steps ("stages" in the FAST terminology) implemented in Java.

3.2 Architecture of the demonstrator
According to the specification of the prototype (Deliverable 1.1), the implementation of the demonstrator is structured into 3 separate modules.

3.2.1 Module 1: Document Retrieval and Entity Extraction
The first module focuses on identifying relevant pages and on extracting interesting entities from them. This is done using what we call an "Second Stage Crawl": we first select relevant entities from an initial set of pages, and then use these entities to generate a new list of pages to be crawled, in order to gather more information related to the identified entities.

The identification of the initial pages is made using a seed generator developed within the project and the crawler integrated in the FAST platform; the seed generator queries a standard search engine using various lists of search words representing different search scenarios that have been considered as relevant to identify various situations related to counterfeiting activities; it produces an initial list of URLs that are used as starting points for a customizable crawl that generates the set of documents from which the information on counterfeiters is extracted.

The relevant pages are downloaded and stored. They are processed using the FAST platform, in order to extract all relevant entities and augmented with information coming from additional sources (such as the WHOIS servers).

Based on ranking algorithms relying on the frequency and popularity of the pages and entities, the most promising entities are selected to generate new seed lists to be fed into the crawler to
download other new pages, not directly related to the initial search scenarios. These pages are also processed using the FAST platform, and added to the document collection. Once all documents have been processed, they are output to:
- A MySQL database that contains all the information regarding the extracted entities;
- A Data Set where each document is represented by an XML file containing the information most relevant for the subsequent modules.

3.2.2 Module 2: Document Clustering
The Data Set resulting from Module 1 consists of a large set of Web pages (10,000 ~ 100,000), each of them associated with a set of extracted entities. These entities can of course be shared by several documents, and one of the important working hypotheses of this project is that documents referring to the same activities or individuals will share a larger proportion of their entities, thus providing a way to cluster them together and extract more exhaustive information from each of the clusters.

To cluster documents that refer to a single activity, website, etc, we represent each of the documents by the set of entities it contains, each of these entities being associated with a weight representing the importance of the entity for the document.

Once this is done, we use a standard clustering algorithm that produces the document clusters based on a similarity between their entity vectors.

3.2.3 Module 3: Case Frames and Results output
In the current version of the demonstrator, Module 3 only displays the produced clusters. The goal is to use the display and “drill down” functionalities built in within the FAST platform in order to help the user to:
- Select the clusters that look most promising, e.g. according to the entities they share.
- Extract from such clusters the information relevant to the final goal of the process.

In this section we describe in detail the implementation of module 1. As stated above, in this first step, the demonstrator receives as an input a set of initial search words – mainly, the name of the affected brands and some keywords related to counterfeiting. The output of this step is a collection of processed and tagged documents, with specific emphasis on the Entities extracted from them.

Across this description, we will include references to one of the tests we have run with the most recent version of the system, in order to illustrate the way the data is processed. This example will be presented boxes similar to this one.
4.1 Data Flow

The following model (Figure 1) is a schema that illustrates how the information flows through the different stages in module 1.

Figure 1
4.2 Generation of a Seed List

A SeedGenerator module has been fully implemented in Java, providing the following functionalities:

- Receiving a seed string (one or several search words), it produces the list of the \( n \) most relevant web pages containing the search words, by querying a large scale web search engine (Google in our case), and outputs this list to a text file.

- The seed words, number of pages and output file can be configured from the command line or by using a simple graphic interface. (Figure 2 shows a screenshot of the seed generator).

- Customizing the crawler configuration in order to start a crawl from the list of web pages that has been generated.

- Receiving an additional list of seed strings, it produces the list of the \( n \) most relevant web pages for each of these seed strings in a non-interactive fashion. This is used exclusively for the Iterative Search, so no interface is required.

In most of the tests we have run, we have been focusing on the counterfeiting of luxury watches. Examples of seed words that have been used range from the most simple description of the target domain to more complex combinations of several concepts:

- Replica Rolex
- Fake luxury watch
- replica OR fake (rolex OR quartier OR "tag heuer" OR (luxury AND watch))
- rolex OR "tag heuer" OR quartier OR (luxury AND watch) (rip-off OR scam)

Figure 2
Although this module is necessary for the final version of the tool, it has been tested separately, and the rest of the system is independently tested on a controlled data set, in order to compare the results of different runs of the system. This means that the initial set of pages used for our tests was not directly generated with SeedGenerator.

In order to generate the test data set, we used the following procedure:

- We chose a domain on which to run the tests: counterfeited watches, specifically Rolexes.

- We produced 2 initial lists of 10 URLs each, both related to the domain but generated with different set of seed words. These 2x10 URLs were obtained by manually filtering a list URLs obtained from the SeedGenerator, and rejecting those that were considered less promising.

- Using the Topodia tool (http://www.topodia.com/) for finding more relevant pages based on a set of training pages, each list of 10 URLs was further expanded to a list of around 150 "similar" pages.

- Each of these lists was then manually process to label each page as “good” (=relevant to the project) or “bad”, with the goal of:
  > Generating supervised test data to evaluate the quality of the page ranking algorithm by testing it on this data.
  > Using the “good” pages as a starting point for the tests of the rest of the prototype, thus limiting the impact of noise on the final results.

The final list of pages used for the tests contained finally 123 URLs.

4.3 Web Crawling and Document Retrieval

The list of URLs obtained from the SeedGenerator, as detailed on section 4.2 above, is fed to a Web Crawler (see below), together with a configuration file. After several test runs, we have come to the following decisions regarding configuration of the crawler:

- The initial run should crawl only one level away from each initial URL. This turned out to be enough to gather most of the relevant information from a website, as the information we are looking for tends to appear in sections of the website that are easily accessible:
  - Contact information of the target website.
  - Product information (products and prices) on the website.

- The second stage crawl is run with a 0-level depth – that is, only the selected URL is used, without any further crawling. Indeed, in this case, deeper crawling further proved to substantially increase too fast the amount of noise.

- To avoid the crawling process to be blocked by some DNS servers, no more than one query is sent to the same DNS every minute. This is also one of the reasons why extensive crawls can delay the whole process – although the reason to opt for a shallow crawl is mainly based on the reasons mentioned above.

- We are still looking for additional criteria to reject some websites, based on rules operating on the format of the URLs (using Regular Expression, e.g. rejecting top-level domains such as .edu...).

This crawl is being executed using the Enterprise Crawler embedded in the FAST platform. This has the advantage that documents are fed to the processing pipeline as they are being
downloaded. The FAST crawler is configured using the .xml file produced by the SeedGenerator as described above. (See Figure 3).

We launched a crawl on 50 URLs using the configuration described in this section. This produced a total of 6'143 downloaded pages, which were fully crawled (but not fully processed) in around 2 hours.

<?xml version="1.0" encoding="utf-8"?>
<DomainSpecification name="IPPtestF">
  <attrib name="accept_compression" type="boolean">yes</attrib>
  <attrib name="allowed_schemes" type="list-string">
    <member>http</member>
    <member>https</member>
  </attrib>
  <attrib name="allowed_types" type="list-string">
    <member>text/html</member>
    <member>text/plain</member>
  </attrib>
</DomainSpecification>

<section name="cachesize">
  <attrib name="pp" type="integer">1048576</attrib>
  <attrib name="pp_pending" type="integer">131072</attrib>
  <attrib name="routetab" type="integer">1048576</attrib>
  <attrib name="starturis" type="integer">1048576</attrib>
</section>

<attrib name="check_meta_robots" type="boolean">yes</attrib>

<section name="crawlmode">
  <attrib name="fwdlinks" type="boolean">no</attrib>
  <attrib name="mode" type="string">FULL</attrib>
  <attrib name="reset_level" type="boolean">no</attrib>
</section>

<attrib name="cut_off" type="integer">5000000</attrib>

[...]
**Figure 4 - Manual Configuration of the Crawler**
4.4 FAST processing

Once a document has been downloaded, it is entered into a FAST processing pipeline, as illustrated in Figure 1 above. FAST processes each documented with a customized list of processing stages (as can be seen in "Figure 5 - Example of Pipeline content" below) that mainly perform the following tasks:

- Removing html tags and tokenize the document.
- Appending whois information to the document.
- Using various rules and methods to extract each type of entity and records them in the corresponding fields.
- Outputting the extracted Entities to the MySQL database.
- Outputting the document in XML format to a hard drive.

<table>
<thead>
<tr>
<th>Document Processing</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>View Pipeline</strong></td>
</tr>
<tr>
<td><strong>Name</strong>: IPPDemonstrator (webcluster)</td>
</tr>
<tr>
<td><strong>Description</strong>: Pipeline for the Demonstrator</td>
</tr>
<tr>
<td><strong>Pipeline contains 50 Stages in the following order:</strong></td>
</tr>
<tr>
<td>DocInit</td>
</tr>
<tr>
<td>DocumentRetriever</td>
</tr>
<tr>
<td>URLProcessor</td>
</tr>
<tr>
<td>URIEquivalence</td>
</tr>
<tr>
<td>Decompressor</td>
</tr>
<tr>
<td>FormatDetector</td>
</tr>
<tr>
<td>SimpleConverter</td>
</tr>
<tr>
<td>FlashConverter</td>
</tr>
<tr>
<td>PDFConverter</td>
</tr>
<tr>
<td>SearchExportConverter</td>
</tr>
<tr>
<td>LanguageAndEncodingDetector</td>
</tr>
<tr>
<td>EncodingNormalizer</td>
</tr>
<tr>
<td>FastHTMLParser</td>
</tr>
<tr>
<td>TeaserGenerator</td>
</tr>
<tr>
<td>LinkNormalizer</td>
</tr>
<tr>
<td>CrawlerLinkFilter</td>
</tr>
<tr>
<td>LinkFilter</td>
</tr>
<tr>
<td>Tokenizer(webcluster)</td>
</tr>
<tr>
<td>SentenceBoundaryDetector</td>
</tr>
<tr>
<td>IterationControl</td>
</tr>
<tr>
<td>URLExtractor</td>
</tr>
<tr>
<td>IPPURLExtract</td>
</tr>
<tr>
<td>IPPDNSExtractor</td>
</tr>
<tr>
<td>IPPPIPExtractor</td>
</tr>
<tr>
<td>WhoisProcessor</td>
</tr>
<tr>
<td>IPPWhoisEmailExtractor</td>
</tr>
<tr>
<td>IPPWhoisZipExtractor</td>
</tr>
</tbody>
</table>

**Figure 5 - Example of Pipeline content**

4.5 WHOIS querying

The documents we are working with have very different contents and formats, and we depend on the efficiency of our Entity Extraction rules (as detailed in section 4.6 below) to obtain useful information from them.

However, additional information can be access in better formatted form from the WHOIS information registered by the owner of the domain in which a web page is located. In order to...
make this additional information available for the rest of the processing, we are appending the obtained WHOIS information to data structure representing each processed document.

WHOIS querying requires up to two queries to external servers. This means that the processing of the pages can be severely delayed if these servers block our system waiting for their reply. Since the WHOIS information on a DNS is the same regardless of the specific page that is being analyzed, we are storing the reply to each WHOIS query in a separate database, and access this database before querying in order to query external sources only once for each new DNS that we discover.

A limited number of the DNS servers that need to be queried to obtain the WHOIS information do not respond to automatic querying. We are working to solve this problem, but for the moment some of the documents simply cannot be completed with the corresponding WHOIS information.

The actual WHOIS query is being executed using a Python script that is invoked from the corresponding FAST processing stage. This script simply obtains the name of the corresponding WHOIS server out of a predefined list, connects to it and launches the query. Then, if necessary, it extracts from the reply the name of the server that holds the final WHOIS information and sends the same query to it. The information is retrieved in the form of a block of text (see an example of a WHOIS query below).

Although the WHOIS reply follows a roughly regular format, we still need to select from the retrieved text the pieces of information that are useful to our system. Currently, we extract the following (see section 4.6 below for more detail on Entity Extraction):

- **Phone numbers**: We use an extractor based on regular expressions. Due to the size and format of the retrieved WHOIS block of text, we relaxed the initial regular expressions (almost any sequence of 7 to 13 numbers appearing in sequence in the WHOIS information is currently considered a telephone number). We do this because:
  - We have observed no false positives so far, since the information inside the WHOIS has no reason to contain a long sequence of numbers that does not indeed correspond to a phone number.

- **Person names**: We use an extractor based on regular expressions and dictionaries. We also added some extra rules, not based on the format of the entity, but on its position in the text (e.g. after strings such as "Administrative Contact.").

- **DNS names**: We extract the information on the primary and secondary DNS registered in the WHOIS information.

- **Company names and Zip numbers**: We use the same extractors as the ones used to process the content of the retrieved document.

The rule and dictionary based extractors used to process the WHOIS information and the content of the documents are the built-in extractors provided within the FAST platform, with possibly small adaptations mentioned above (see also the section 2.6 below)

---

**Example of WHOIS Query**:

Registration Service Provided By: NameCheap.com
Contact: support@NameCheap.com
Visit: www.NameCheap.com

Domain name: REPLICAHAUSE.COM
Registrait Contact:
The Hause Watches
James Henderson (replicahause@yahoo.com)
+81.54139411
Fax: +1.5555555555
6F, Mizuho Building
2-92-4 Kamiosaki
Shinagawa-ku,, Tokyo 141-0021
JP

Administrative Contact:
The Hause Watches
James Henderson (replicahause@yahoo.com)
+81.54139411
Fax: +1.5555555555
6F, Mizuho Building
2-92-4 Kamiosaki
Shinagawa-ku,, Tokyo 141-0021
JP

Technical Contact:
The Hause Watches
James Henderson (replicahause@yahoo.com)
+81.54139411
Fax: +1.5555555555
6F, Mizuho Building
2-92-4 Kamiosaki
Shinagawa-ku,, Tokyo 141-0021
JP

Status: Locked

Name Servers:
NS1.HAUSESOLUTIONS.COM
NS2.HAUSESOLUTIONS.COM

After full processing, the following Entities where extracted from the above text:
dnsname: replicahause.com
ipaddress: 201.218.228.243
whoisemails: replicahause@yahoo.com
whoisphonenumbers: (+81) 5413-9411; (+1) (555)5555-555
whoispersonnames: James Henderson

The phone number information is clearly useless. However, the other entities look quite promising.

4.6 Entity Extraction and Normalization
Deliverables 1.1 and 1.2 contain a detailed list of the entities we defined as relevant for extraction. The table below describes in more detail how each extraction has been implemented.

<table>
<thead>
<tr>
<th>Entity</th>
<th>Extraction Implementation</th>
</tr>
</thead>
<tbody>
<tr>
<td>DNS name</td>
<td>1. The DNS for each page is obtained from the document's URL.</td>
</tr>
</tbody>
</table>
Due to the large number of exceptions, this is not done using
Regular Expressions but operating on the URL as a String.

2. Additional DNS are obtained from a page's WHOIS
information, using a regular expression that refers to the labels
preceding the DNS name: "Primary DNS." or "Secondary
DNS".

**Phone number**

FAST incorporates a phone number extractor that uses a complex
regular expression definition (defined in .xml), taking into account:
- Usual format of US numbers
- Usual format of international numbers, that identify country and area
code with parenthesis or add symbols.
- Context: Identification of any sequence of digits found near words like
phone, call, etc.

This extractor also provides a normalization functionality that tries to
represent all phone numbers in a similar format – only those without
complete information (when, for example, the country code is missing)
ar are not normalized.

**Company name**

We are using two of the three Company extractors provided by FAST:
- We have ignored the extractor based solely on white list
dictionaries, since we are not interested in top, well-known
companies, but in smaller ones.
- The two extractors we are using combine regular expressions
with a dictionary of company names and a dictionary of
company form suffixes, both in full form and abbreviated (Inc.,
Ltd., AG, etc.).

It is difficult to measure accurately the recall of this extractor, since this
would require a manual extraction of the target company names across
the data set. From the analysis performed through random sampling,
we have come to the conclusion that a large number of company
names that can be identified by the human eye are being ignored by
this extractor. Nevertheless, we are currently not focused in improving
this, since its impact on the quality of the final results is yet to be
determined.

**Person name**

We are currently using the extractors provided by the FAST FAST
platform, that are based on:
- A dictionary of common first names
- A dictionary of common surnames
- A set of rules on capitalization and abbreviation.

We have added a stage that removes famous names:
- Names of famous people being used with marketing purposes
- Brand names

We had considered normalizing names in order to combine different
instances of the same name with slight spelling differences. We have
come to the conclusion that it will be easier to add some flexibility to
the algorithm that compares names for clustering purposes in module
2.

**E-mail addresses**

E-mail addresses are being extracted using a trivial regular expression
based on the presence of the @ symbol.

**Product name and Price**

In order to implement a reliable product and price extractor, we depend
on the ability for our techniques to focus on product names, which
could for example be explicitly provided by the company interested in
using our tool.
The reason for this is that product names are too diverse, and even
using a price tag as an anchor, it remains too difficult to identify what
words close to the price tag form a product name.
Using a limited list of target products, we can look for them in the page
and then identify each of them with its price tag, using a mixed criteria
based on distance or surrounding words.
This is currently not being extracted.

| Physical address | This is currently not being extracted, as the WHOIS information is not
|                  | formatted strictly, and thus separating the address from the rest of the
|                  | contact information is not a trivial task.
|                  | Zip numbers are being extracted using a standard tool from the FAST
|                  | platform, that refers mainly to the US zip format.
| Payment Method   | This is currently not being extracted.

After full processing of 19'856 different documents, Module 1 extracted the following
amounts of various entities:
25744 distinct e-mails
5915 distinct IP addresses
149007 distinct person names
196208 distinct phone numbers
14911 distinct company names
6990 distinct DNS names

4.7 Document and Entity Output
In order to be able to test each module of the prototype separately, it was important to make the
results of module 1 available as a independent input for module 2. To do so, we used the
following procedure:
- Outputting all relevant entities to a MySQL database. This allows the Scoring and
  Ranking functions to work more efficiently by directly accessing to this information.
- Outputting each document to an XML file that contains all fields that will be necessary for
  modules 2 and 3.

<?xml version="1.0" encoding="utf-8" ?>
<documents>
  <document>
    <element name="documentid">
      <value><![CDATA[http://www.newwatch4u.com/]]></value>
    </element>
    <element name="body">
      <value>[...]</value>
    </element>
    
    <element name="whoispersonnames">
      <value><![CDATA[Carl Anders; Carl Anders]]></value>
    </element>
    <element name="whoisinfo">
      <value>[...]</value>
    </element>
  </document>
</documents>
Figure 6 - Example of an .xml file representing a document

At the end of Module 1, we produced 20'015 (not necessarily distinct) .xml files, that took 471 MB of disk space.

The difference between the number of .xml files produced and the number of distinct documents processed by Module 1 (19'856) is due to the fact that the First Crawl and the Second Stage Crawl are run separately making some of the documents appear in the output twice.

4.8 Page Scoring and Entity Selection

After all documents in the first run have been processed, a large list of potentially relevant entities is available. The goal is then to choose which entities seem promising enough to be used for the second stage crawl. This is done by scoring the pages and the entities, in two steps:

- **Page Scoring**: The goal is to define a set of criteria to reject pages that should not be considered for the second stage crawl (although they are still taken into account for clustering in Module 2).

  In order to explore possible criteria to decide what pages to reject, we created a Test Set with our Seed Generator and manually categorized each page into either "good" or "bad". We then tested several criteria on this set, and compared the obtained result to the result of the manual ranking, computing recall and precision scores.

  As a result of this analysis, we found that the best results were obtained by discarding the pages that were located on a website whose domain name had a popularity in the whole internet (i.e. the numbers of web pages containing them, as provided by Google) above a given threshold. We thus came to the conclusion that it could be interesting to make and
use the hypothesis that very popular websites have a stronger commitment to verify the content of their sites and would not allow for hosting counterfeiting activities.

Notice that this page scoring method is only valid for the pages resulting from the first crawl, where we expect all pages to refer to actual counterfeiting activity.

Currently, our demonstrator uses the rule that websites whose domain names have popularity in Google greater than 11000 hits should not be considered for the second stage crawl.

- **Entity Scoring:** Once we have filtered out the pages that are not promising enough, we want to select that entities should be used for the second stage crawl. Defining good criteria to do this is a more complex problem, since it is difficult to know a priori whether an entity is good or bad, in particular without analyzing its impact in the final results of the prototype.

Based on our experience with Page Scoring, we decided to explore similar criteria to define entity eligibility. From a conceptual point of view, this might seem reasonable, since:

- We do not expect counterfeitters to have a widespread presence on the Internet
- Rejecting precisely the entities that produce most of the hits might also reduce the noise introduced in the System.

Currently we keep all entities that appear in a relevant page and have a Web frequency (as provided by Google) below a threshold that depends on the type of the entity.

- **Implementation:** The code that performs page scoring and Entity Selection has been implemented in Java. It works in the following way:

  - It first accesses the MySQL database to retrieve the list of processed documents.
  - For each document, it verifies the frequency of the DNS name and ranks it as either "good" or "bad".
    - All frequencies are stored in a separate database, in such a way that Google needs to be queried only once for each entity. Otherwise, the amount of generated network traffic to Google would become excessive.
  - It accesses the MySQL database to retrieve the list of entities of each type.
  - For each entity that appears in a relevant document, its global frequency is compared to the threshold that corresponds to the entity type. If the entity is sufficiently uncommon, it is added to the list of entities to be refed. In addition, the DNS name of a "good" page is always taken into account for the second stage crawl.

After applying the different scoring functions, 534 entities were selected chosen for being used for the second stage crawl.

**4.9 Second Stage Crawl**

Once the process described in section 0 is completed, a list of entities is available to be used for the second stage crawl. This list is fed into the SeedGenerator module, in order to obtain a new list of URLs. For the second stage Crawl, only the top 10 hits for each entity were retrieved, in order to keep the amount of processed data manageable.
The SeedGenerator produces a file with all the URLs to be crawled. A new collection is automatically created in FAST, with this file as starting point. As already mentioned, the crawl is set to 0-level depth (that is, we do not crawl away from the URLs in the list), and the new collection is processed through steps 4.3 to 4.7.

At the end of this process, complete information for all documents from the initial collection and from the second stage Crawl is stored in both the MySQL database and in the form of XML files.

A list of 16'612 URLs was generated by the second stage Crawl. This resulted in 13'952 new documents being processed and added to the collection. The crawling of these documents took less than 2 hours.

The reason why this crawl took less than the initial one is that there is a delay in the crawler between each request to the same website. This means that crawling inside a website takes longer than crawling URLs from different websites, even if this number is larger.

5 Implementation of Module 2 – Document Clustering

Module 2 starts with the information generated by Module 1:

- A MySQL database with information on the extracted entities.
- A data set with each document represented by an XML file.

Based on this data, the goal is then to identify clusters of documents, sharing most of their entities. To do this, we use the unsupervised clustering functionality built in the FAST platform.

The following chart (Figure 1) illustrates the data flow within this module.
Figure 1: Module 2 data flow

For the clustering, each document is represented by a vector (hereafter called a docvector) containing the entities that have been extracted from it, along with a weight between 0 and 1 measuring the importance or reliability of the entity.

The clustering is then performed based on similarities between docvectors.

5.1 Generation of the docvectors

The docvectors summarizing the information in each document are created by a Java application that:

- Retrieves the list of the documents in the collection from the MySQL database.
- For each document, obtains the list of entities that were extracted from it.
- Creates a docvector, adding to each entity a weight, based on:
  - The type of entity
  - The entity score (the inverse of its global frequency in the Web).
- Stores the docvectors in a separate table in the database.

The page [http://www.italianfakes.net/](http://www.italianfakes.net/) was summarized by the following docvector:

```
[italianfakes.net,1][202.157.175.196,1][310 721-9474,1.0][italianfakes@yahoo.com,1.0]
```

And the corresponding .xml file was:
5.2 Reloading documents

Once the docvectors have been created, a new collection is created in FAST. This collection is built from the directory where all the xml files are stored, using the FAST File Traverer.

The FAST File Traverer is a tool that allows previously stored documents to be fed into the FAST pipeline without using a crawler. Since all documents produced in Module 1 where output in XML format in a separate folder, we only need to instruct the File Traverer to scan the suitable folder and upload each document back into the new collection. The XML format allows avoiding reprocessing the documents: all entities that where extracted are collected directly into their corresponding containers.

Each reloaded document is imported into FAST to be further processed. At this point, the FAST pipeline includes only one stage: importing the docvector from the MySQL database.
5.3 Document clustering

Once all documents have been loaded, we use the FAST clustering tool to generate the clusters. The unsupervised clustering process works as follows:

- In a first phase, all documents are retrieved from the Index, together with their document.
- In a second phase, the initial cluster tree is produced. The N document vectors are processed sequentially using a "follow-the-leader" algorithm, where document \( i \) ends up in cluster \( k \) if:
  - document \( j \) (where \( j < i \)) is in cluster \( k \) AND
  - similarity(\( i, j \)) >= threshold AND
  - \( j = \text{argmax}_j(\text{similarity}(i, j)) \).

  Otherwise \( i \) is put in its own cluster. In other words, the algorithm places \( i \) in the best existing cluster, or creates a new cluster if no existing cluster is good enough. At the end, singleton clusters are removed. The similarity used for clustering is the cosine similarity.

- In a third phase a labeling algorithm is applied to the cluster tree. The labels for a cluster are produced from the strings in the document vectors of the cluster members.
- In a final phase, a "label unification" process is carried out, where the system tries to make the computed label sets more unique (by removing elements that might be in use several places). Both sibling nodes and parent/child nodes are handled.

Several configurable parameters are available to control details the clustering process. We are currently working on the threshold and the weights used for the entities in order to improve the results.

5.4 Cluster labeling

Using the FAST Content API, the name of the cluster is added as a field to the representation of each of the documents it contains. In such a way, these fields are available for browsing with the Search View tool in Module 3.

In the example we have been following across this document, the following 3 clusters were identified:

Cluster name: <fineluxurytime.com.cn;no ip resolved>
- document: http://www.gopui.com/
  
Cluster name: <213.219.245.36;204.16.194.104;ireplicastore.com>

Cluster name: <209.34.176.11;michelle chen>
These three clusters were manually analyzed, with the following conclusions:

1. Cluster 3 identifies a counterfeiter. Two different websites have been identified (watchec.com and rolex-replica-watch-shopping.com). They correspond to the same IP address, and the name of michelle chen appears in both WHOIS descriptions. Therefore, further investigation on this name should lead to a potentially good case.

2. The reason for the number of clusters to be so low is that the docvectors currently contain (too many entities with too high weights). Because of this, only the documents with the shorter docvectors could be clustered together. Additional Work on improving the selection of Entities for the docvector should lead to immediate increase of the number of produced clusters.

6 Module 3 - Results output

In the current implementation, the clustering process is fully performed in the FAST. This means that the produced cluster information can be subsequently accessed with the FAST Search View tool. As shown in the Figure 7 below, the FAST navigators allow for a specific cluster to be chosen, and the information on the entities present in the documents in that cluster to be displayed.
7 Tests and Evaluation

After evaluating the results produced by the different stages, we came to the following conclusions:

- Module 1 is working properly: The document selection is correct, and the entities are being extracted with a high recall. The number of entities extracted show that a lot of information to work on.

- Entity extraction precision still needs to be improved: Currently many entities that will not be useful in the final process are still extracted. The reason for this is because:
- The presence of false positives (for instance, Taj Mahal is being extracted as a person name).

- Some correctly extracted entities are actually irrelevant to the global process (for instance, names of famous personalities or companies)

- Module 2 is completely implemented, and has shown good potential as it was able to generate relevant cases even with a very simple configuration. Further work is required in order to improve the quality and quantity of extracted clusters.

- The prototype is working as expected in terms of efficiency and scalability: given the amount of the documents and processing tasks performed on them, it was able to generate results in a manageable time. The fact that system requires some manual controls however implies that multiple tests cannot be run in parallel to analyze (automatically) different configurations, but this will be solved in the next release.

8 Future Work on Implementation

We have already identified several aspects that will require improvements to be made for the final version of the prototype.

Regarding Module 1, the following issues need to be worked on:

a. **Whois**: The module that retrieves the whois information can block the document processor while it is waiting for an answer from server. We are already reducing this by storing in a database all whois reports retrieved in the past, but it should be further improved.

b. **Entity selection**: The selection of Entities to be refed should be more intelligent. Currently, all entities that are below the "popularity" threshold are refed. This causes the iteration to be too exhaustive, which means that there is potentially a lot of noise, and also that it is difficult for us to compare the results.

c. **Entity scoring for the docvectors**: In this initial run, we used the simplest way to create the docvectors and decided to keep all the extracted entities passing the popularity filter in the docvectors. A more sophisticated entity filtering procedure should be used. Possible ideas to be taken into account for scoring are:
   - Using information on the frequency of the entity across the data set.
   - Taking into account the position of the entity in the document

d. **Complete architecture**: In the current implementation, human control is required to monitor the transit from one module to the next. We are working on a way to implement the whole system as one single tool that can be run automatically, but there are currently remaining issues related with the interaction between Java and FAST that prevent this from being completely reliable.

Regarding Module 2, we are conscious that the similarity clustering might be not the best approach for our goal. As detailed in the initial project description, the final goal is to implement this module using Link-Based Analysis. We are currently working on the development of a new version of the modules, but it has to be noticed that the current implementation, even much simpler, already proved to be useful to provide interesting results.

Regarding Module 3, a specific interface will be developed that fills visualized the extracted information in a more suitable way with additional browsing functionalities.

---

1 Raymond J. Mooney and Un Yong Nahm. “Text Mining with Information Extraction”. 

S. Voss and C. Joslyn. “Advanced Knowledge Integration in Assessing Terrorist Threats”.


Jaime A. Reinoso Castillo et al. “Information Extraction and Integration from Heterogeneous, Distributed, Autonomous Information Sources – A Federated Ontology-Driven Query Centric Approach”.