Acquisition of Categorized Named Entities for Web Search

Marius Paşca
Google Inc.
1600 Amphitheatre Parkway
Mountain View, California 94043
mars@google.com

ABSTRACT

The recognition of names and their associated categories within unstructured text traditionally relies on semantic lexicons and gazetteers. The amount of effort required to assemble large lexicons confines the recognition to either a limited domain (e.g., *medical imaging*), or a small set of predefined, broader categories of interest (e.g., *persons*, *countries*, *organizations*, *products*). This constitutes a serious limitation in an information seeking context. In this case, the categories of potential interest to users are more diverse (*universities*, *agencies*, *retailers*, *celebrities*), often refined (e.g., *SLR digital cameras*, *programming languages*, *multinational oil companies*), and usually overlapping (e.g., the same entity may be concurrently a *brand name*, a *technology company*, and an *industry leader*). We present a lightly-supervised method for acquiring named entities in arbitrary categories. The method applies lightweight lexico-syntactic extraction patterns to the unstructured text of Web documents. The method is a departure from traditional approaches to named entity recognition in that: 1) it does not require any start-up seed names or training; 2) it does not encode any domain knowledge in its extraction patterns; 3) it is only lightly supervised, and data-driven; 4) it does not impose any a-priori restriction on the categories of extracted names. We illustrate applications of the method in Web search, and describe experiments on 500 million Web documents and news articles.

Categories and Subject Descriptors

H.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*search process*; H.3 [**Information Storage and Retrieval**]: Online Information Services—*Web-based services*; I.2 [**Artificial Intelligence**]: Natural Language Processing—*text analysis*; I.2 [**Artificial Intelligence**]: Learning—*knowledge acquisition*

CIKM'04, November 8–13, 2004, Washington, DC, USA.

General Terms

Algorithms, Experimentation

Keywords

Web information retrieval, lightweight text processing, named entity extraction, related names and categories, information integration

1. INTRODUCTION

1.1 Problem

Within the general goal of information retrieval - finding the documents that are relevant to a user's information need - Web retrieval must cope with additional complications in terms of user input and output. On the input side, users submit queries that usually contain a small number of words. On the output side, there are quantitative limits since most users actually inspect only the first few documents in the search result set [15]. The combination of these two constraints makes the Web retrieval problem analogous to finding the proverbial needle in the haystack, where the haystack consists of billions of items (Web documents) and the needle is a set of a few (most-relevant) documents.

When looking closer at the unstructured text in documents and users' queries, it is apparent that terms and phrases are not all equal. Beyond measurements like document frequencies or term proximity, the occurrence of *named entities* signals prominent pieces of information. Often, users will search for lists of names through queries such as *Middle Eastern countries* (or *Islamic nations*), *races* (or *sports events*), *movies* (or *home video releases*). Even more often, searches refer to popular names such as *Paris*, *Kentucky Derby* or *Harry Potter*, as indicated by the frequent occurrence of such names among the top search engine queries. In other cases, users might search for unfamiliar rather than popular names to address more basic information needs, e.g. figuring out what kind of name they are dealing with (could it be a *reptile*; a *modern programming language*; or a *Greek dragon*?). As illustrated in Figure 1, our haystack of documents can be then considered from a different perspective: that of a goldmine that "hides" valuable information nuggets about various names, including nuggets that encode their categories.

Traditionally, the recognition of names and their associated categories within unstructured text relies on semantic lexicons and gazetteers. The amount of effort required to assemble large lexicons sometimes confines the recognition

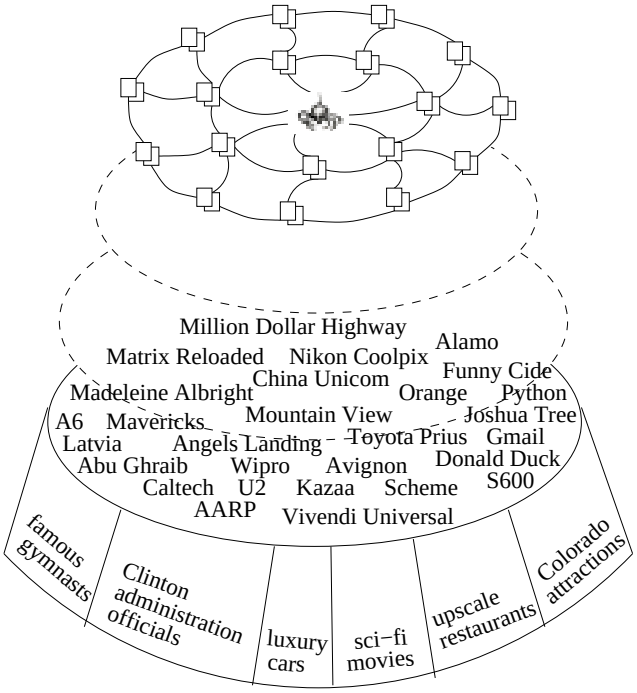

Figure 1: Names (middle layer) and their categories (lower layer) are hidden within the unstructured text of Web documents

to a limited domain (e.g., *medical imaging*) for which large-coverage resources already exist. Alternatively, the recognition is limited to a small set of broader, pre-defined categories of interest, e.g. *persons*, *countries*, *organizations* and *products*. This latter alternative has become the de-facto standard after its introduction in the Message Understanding Conference [6]. For many information and language processing tasks, such a coarse-grained set of categories is appropriate. It may become a serious limitation, however, in an information seeking context, especially in Web search. In this case, the categories of potential interest to users are more diverse (*universities*, *agencies*, *retailers*, *celebrities*) and more refined (e.g., *SLR digital cameras*, *programming languages*, *multinational oil companies*). Moreover, the categories are usually overlapping since the same instance may be concurrently a *brand name*, a *technology company*, a *storage provider*, and an *industry leader*. Ideally, the target instance should be retrieved consistently regardless of which of the legitimate categories is used in a particular query.

1.2 Approach

In this paper, we present a lightweight, lightly-supervised method for acquiring named entities in arbitrary categories from Web documents. In the trade-off between method complexity and output coverage, we opted from the start for low complexity. The method is purposely designed to be simple, and thus handle robustly the noise and diversity of Web documents. We focus on textual content rather than structural clues. Shallow lexico-syntactic extraction patterns are applied to the unstructured text of Web documents and Web news articles. Overall, the method is a departure from traditional approaches to named entity recognition in several respects. First, there is no need for training that would provide extraction rules or knowledge about specific categories of names. Second, the method does not require any initial clues or seed names, which would otherwise have to be specified for each category. Third, the extraction patterns do not encode any domain knowledge, which avoids any restriction to a given domain. Furthermore, the method is only lightly supervised, and data-driven. Lastly, it does not impose any a-priori restriction on the categories of extracted names.

The remainder of the paper is structured as follows. Section 2 illustrates the process of derivation of categorized names from unstructured text. The categorized names can be loosely integrated into existing knowledge resources as shown in Section 3. They also enable several Web search applications, which are described in Section 4. Section 5 presents evaluation results for the extraction method on 12 million Web news articles and 500 million Web documents. After further discussion in Section 6, we glance at future work in Section 7.

2. EXTRACTING CATEGORIZED NAMES

In this section, we introduce a lightweight method for identifying names and their categories within unstructured text. The input consists of a small set of domain-independent, lexico-syntactic patterns. The output is a set of names with their corresponding categories as derived from arbitrary Web documents. As shown in later sections, these kinds of simple methods, when applied to large amounts of data, can open the path towards novel applications to Web search.

2.1 Lightweight Extraction Method

The extraction is a three-step process consisting of document pre-processing, extraction of categorical facts, and derivation of categories. To ensure robustness on large collections, the extraction relies on minimal tools and resources.

2.1.1 Document Pre-Processing

After filtering out HTML tags, the input documents are tokenized, split into sentences and part-of-speech tagged using the TnT tagger [2]. The tags are only used in the last step to derive the name categories. Languages such as English distinguish proper names from other nouns through capitalization. Therefore each sequence of capitalized terms in the sentence is marked as a potential instance name.

2.1.2 Extraction of Categorical Facts

The presence of potential instance names and simple lexico-syntactic patterns in sentences, inspired by [14], is interpreted as a signal of a categorical fact associated with the name. A categorical fact is a sentence nugget that is likely to provide explicitly the category of the associated instance name. The fact and associated instance name are captured with a set of patterns which can be summarized as:

⟨[StartOfSent] X [such as|including] N [and|,|.]⟩,

where N is the potential instance name and X is the categorical fact. The matching of the patterns in the sentences results in pairs (X,N) of categorical facts and instance names, which are underlined the sample sentence *"That is because software firewalls, including Zone Alarm , offer some semblance of this feature"*. All potential instance names that are not associated with a categorical fact are discarded.

Table 1: Selection of categories from categorical facts

Categorical fact and instance name	Selection
Anti-GMO food movements sprouted up in European nations in the 1990s, including Germany	Discard
Our customers' chipsets compete with products from other vendors of standards-based and ADSL chipsets, including Alcatel	Discard
The venture is supported by a number of academics, including Noam Chomsky	Retain (*academics*, *Noam Chomsky*)
API Adapter can be written in other programming languages such as C++	Retain (*programming languages*, *C++*)

2.1.3 Derivation of Data-Driven Categories

In the final step, the categorical facts X of the pairs (X,N) are searched for the noun phrase which encodes the category of the associated name. This phrase is approximated by the rightmost non-recursive noun phrase whose last component is a plural-form noun. While such a coarse approximation would cause certain complex categories to be missed on small collections, it is more scalable to millions of Web documents since it avoids complications related to deeper text understanding, e.g. prepositional phrase attachments. One of the following situations may occur as illustrated in Table 1: 1) No plural-form noun phrase exists near the end of the categorical fact; the pair (X,N) is discarded; 2) A plural-form noun phrase exists near the end of the categorical fact, but it is immediately (e.g., within 5 tokens) preceded by another plural-form noun phrase; the pair (X,N) is discarded; 3) Otherwise, the noun phrase is retained as the lexicalized category of the instance name N. Note that one of the categories selected in Table 1 is *programming languages* rather than *other programming languages*. Non-informative adjectives like *other*, *several*, or *many* are among the Top-20 most-frequent modifiers computed statistically in a post-processing phase over the entire set of categories.

2.2 Extensions

2.2.1 Extraction of Similar Names

Textual enumerations offer an inexpensive way of extracting multiple, rather than single, categorized names from a given categorical fact. To support multiple-name extraction, the patterns are slightly modified to match enumerations (N_1[,N_2,... and N_m]) in addition to single names (N).

2.2.2 Automatically-Derived Patterns

Even though the basic method is mostly unsupervised and does not require any clues to start up, it does rely on manually-selected patterns. A logical extension is the identification of additional patterns to increase coverage, i.e. the number of categorized pairs.

Figure 2 illustrates a statistical method for acquiring new extraction patterns from unstructured text. The idea is to match pairs (C, N) of categories and names, extracted in the previous iteration, back into text sentences. Each potential pattern has the form:

$\langle LeftContext$ C $InnerPattern$ N $RightContext\rangle$,

where $LeftContext$, $InnerPattern$ and $RightContext$ represent contiguous sequences of sentence terms. The entire

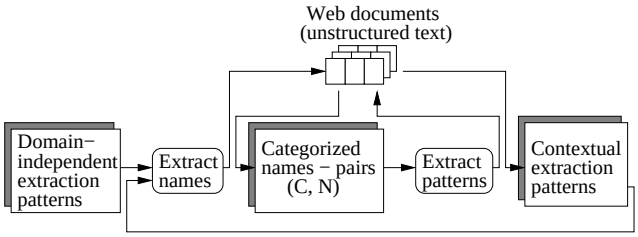

Figure 2: Acquisition of extraction patterns from unstructured Web documents

pattern is completely contained within a sentence. To avoid any bias towards high-frequency pairs, duplicates of potential patterns that occur for the same category C and name N are discarded. Table 2 shows the Top-15 patterns ranked according to their frequency. In this experiment, $LeftContext$ and $RightContext$ are defined by the part of speech tags of at most 5 terms in the same sentence. Comparatively, $InnerPattern$ is defined by the actual words and their tags. The tags are from the Penn Treebank [17] tag set, e.g. *NNP* is a proper noun, *CC* is a conjunction etc. The relative position of C and N is denoted in Table 2 by the superscript ^{a} if N is before C, and ^{b} otherwise.

Table 2: Ranked contextual patterns acquired from text

$LeftContext$ (*POS tags*)	$InnerPattern$ (*words*)	$RightContext$ (*POS tags*)
StartOfSent	such as^b	, NNP NNP , NNP
, NNP , NNP ,	and other^a	. EndOfSent
IN NNP , NNP ,	and other^a	. EndOfSent
StartOfSent	such as^b	, NNP NNP CC NNP
NNP NNP , NNP ,	and other^a	. EndOfSent
StartOfSent	such as^b	, NNP CC NNP VBP
StartOfSent	such as^b	, NNP , NNP ,
StartOfSent	, including^b	, NNP NNP , NNP
StartOfSent IN	such as^b	, NNP NNP , NNP
StartOfSent DT	include^b	, NNP , NNP ,
StartOfSent	, including^b	CC NNP NNP NNP NNP
NNP , NNP NNP ,	and other^a	. EndOfSent
StartOfSent JJ	, including^b	CC NNP NNP , VBP
StartOfSent JJ	, including^b	, NNP NNP CC NNP
StartOfSent DT	are^b	, NNP , NNP ,

The results in Table 2 show that, in addition to "recovering" the initial $InnerPatterns$ (*"such as"* and *"including"*), the top automatically-derived patterns also "uncover" other useful matches (*"and other"*; *"include"*; and *"are"*). The process continues with a next iteration, in which the new set of patterns generates an expanded set of categorized names, which in turn can be used iteratively to generate another set of potential patterns, as suggested in [4, 23].

3. CATEGORIZED NAMES AS LEXICAL KNOWLEDGE

Each categorized name corresponds to an $InstanceOf$ assertion between the name and the category, where the latter is a lexical concept. This section describes the use of the assertions to find related categories, and also to expand existing knowledge resources.

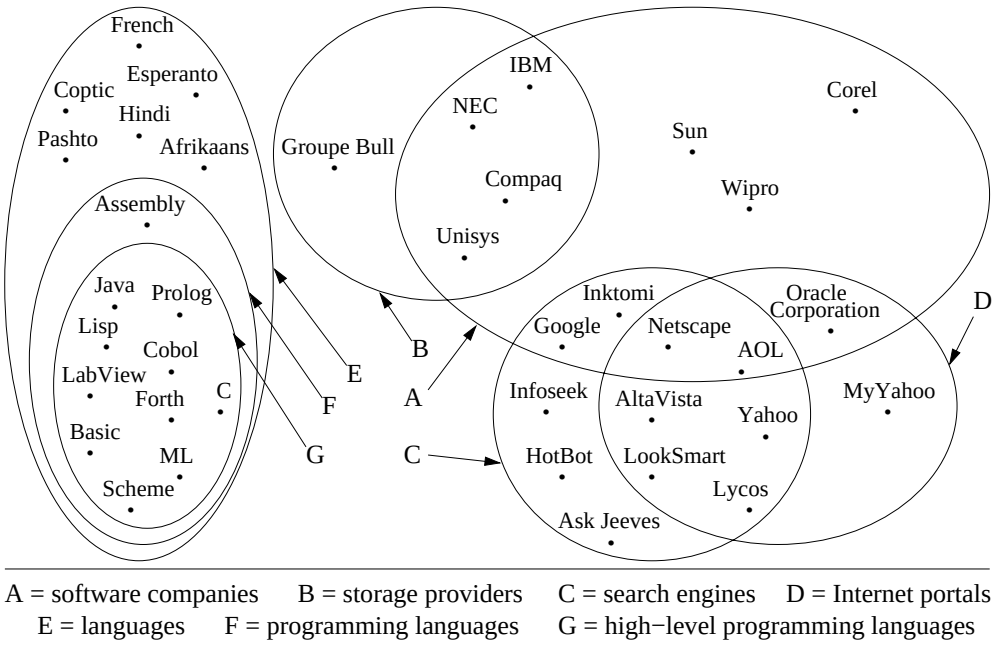

Figure 3: Instance set overlap indicates related categories

3.1 Category Relatedness as Set Overlap

Ideally, an IsA semantic relation between two categories will be reflected in their sets of instance names, one of which will completely include the other. However, in practice the instance name sets are partial rather than complete. Their overlap, or lack thereof, is no longer a pure reflection of the inter-category relations. On the positive side, overlaps between two sets still indicate a strong relation between the corresponding categories. In fact, set overlap will sometimes reveal something that formal knowledge resources may fail to capture, i.e. category relatedness.

Figure 3 illustrates categories identified as related based on their set overlap. All categorized names were actually extracted from Web documents, even though only a subset of them is included in the figure for clarity. Empty or very small overlaps correspond to categories that are not directly related. [1] This is the case for *languages* and *search engines*, for instance. Medium or large-sized overlaps indicate related categories, e.g. *search engines* and *Internet portals*, or *search engines* and *software companies*. In some cases, the overlap is complete in the sense of set inclusion. The corresponding categories are classified one under the other, e.g. *high-level programming languages* and *programming languages*, or *programming languages* and *languages*. But how does this compare to what other knowledge resources encode already? WordNet [11] is among resources that have been widely studied in the context of information retrieval [28, 12]. WordNet is a lexical database which organizes English lexical concepts along several semantic relations. For our example, WordNet explicitly encodes that a *programming language* is a subconcept (or hyponym, in WordNet terminology) of *language*. However, WordNet does not encode any of the other relatedness relations mentioned above. In fact, *software companies*, *Internet portals* and *high-level programming languages* do not have any corresponding concept in WordNet. This makes the categorized names a useful addition to various knowledge resources, as discussed in the following.

3.2 Integration into Existing Knowledge Resources

An immediate extension to any knowledge resources that organize English concepts hierarchically is to map (some of) the extracted names into new $InstanceOf$ assertions. In the case of WordNet and other similar resources, the new assertion corresponds to a new instance node being linked to an existing node at the bottom of the hierarchies. For example, the categorized names *Google* (in the category *search engines*), *Swiss National Bank* (in *central banks*), *Joschka Fischer* (in *foreign ministers*) and *State Farm* (in *insurance companies*) each generates a new leaf node inserted under the WordNet concepts *search engine*, *central bank*, *foreign minister* and *insurance company* respectively. The process is straightforward only if there is exactly one possible insertion point, i.e. the name belongs to exactly one category, the category matches exactly one WordNet concept (after reduction of the category to its base, singular form), and the latter is a leaf node. [2] In the general case, however, the name belongs to multiple categories, each of which matches zero, one, or multiple WordNet concepts. For example, the category *European companies* does not match any WordNet concept, whereas the category *rappers* matches two WordNet concepts (one with the sense of artists, the other with the sense of knocking devices).

The conservative insertion algorithm shown in Figure 4

[1] Instance names that concurrently denote objects of different types, e.g. *Orange* (in *companies*, *counties*, *cities*) and *Taj Mahal* (*famous buildings* and *musicians*), produce small overlaps that should be ignored.

[2] A WordNet concept is considered a leaf node if it is situated towards the bottom of the hierarchy, i.e. some or all of its immediate existing hyponyms are names.

Input: a name N and its categories C_i, i=$\overline{1 \ldots n}$
Output: A hierarchy concept to link N to, or $None$
Variables: $\{W\}$ = set of pairs ⟨hierarchy concept, count⟩
Steps:
1. $\{W\}$ = empty set
2. For each category C_i of the name N
3. Match C_i into hierarchy concepts $\{W_i\}$
4. If $\{W_i\}$ is empty
5. Discard a modifier of C_i and goto 3.
6. For each matching hierarchy concept W_i
7. For all hyponyms W_j of W_i, including W_i
8. If W_j is a leaf concept
9. Find entry for W_j in $\{W\}$
10. If no entry found
11. Insert entry (W_j, 1) in $\{W\}$
12. Else
13. Increment counter of entry for W_j in W
14. Find concepts $\{H\}$ of $\{W\}$ with the highest count
15. If $\{H\}$ has one element
16. Return element of $\{H\}$
17. Else
18. Compute least-general concept L of elements $\{H\}$
19. If L exists and is a leaf concept
20. Return L
21. Else
22. Return $None$

Figure 4: Algorithm for inserting categorized names into hierarchical knowledge resources such as WordNet

links a name to at most one WordNet concept. [3] Initially, the algorithm matches each category of the input name against WordNet concepts. If a category does not match any WordNet concept, its modifiers are discarded until one or more matches are found. Thus, *high-level programming languages*, *Internet portals* and *science fiction writers* match the WordNet concepts *programing language*, *portal*, and *writer* respectively. For each explicit match into a WordNet concept, there is an implicit match into each of its hyponyms, i.e. concepts in the hierarchy rooted at the matching WordNet concept. For instance, the explicit match of *languages* to *language* entails implicit matches to *artificial language*, *natural language*, *Indo-European language* and so forth. The algorithm selects as insertion point the WordNet leaf concept with the highest number of matches over all categories. In case of ties, the least-general unifying concept is selected.

The algorithm in Figure 4 introduces only $InstanceOf$ assertions via the inserted instance names. The hierarchical structures are otherwise preserved. The alternative is to create intermediate concepts as well, which may be missing due to the fact that WordNet is a general-purpose resource that does not aim at fully representing specialized domains [11]. Intermediate concepts, e.g. *software company*, *high-level programming language* act as a middle layer between existing concepts (*companies*, *programming languages*) and newly acquired names. Their introduction brings additional complications in terms of consistency (is a *search engine* a kind of *engine* or rather a kind of *company*?) and coverage (what other concepts besides *high-level programming language* form a partition over *programming language*?). On the other hand, the simplifying factor in tackling intermediate concepts is precisely their reliance on the sets of names acquired from Web documents. Each name in the category is equivalent to an anonymous, democratic vote that the category actually corresponds to a useful concept that collectively represents the properties of its members. From this point of view, *red car* or *tall company* are less useful or improbable concepts since the Web as a decentralized knowledge repository provides little or no evidence for them. [4]

Besides $InstanceOf$ assertions, category relatedness represents another piece of knowledge to integrate in other resources. While less formal and reliable, the knowledge that *search engines* and *Internet portals*, or *storage providers* and *companies* are related is certainly helpful. Each pair of related categories is represented as an additional $RelatedTo$ link in the existing hierarchical structure. Resources enriched with $RelatedTo$ links become more useful in linguistic-oriented applications that search for loose rather than strict relations among pairs of concepts. For example, any application computing lexical chains [12, 13, 27] among words, phrases or concepts can use the $RelatedTo$ links as the main raw material in the identified lexical chains. The $RelatedTo$ links are equally useful for query expansion, or suggestions for query refinement.

4. APPLICATIONS TO WEB SEARCH

The categories of names acquired offline from the Web offer an alternative view of the underlying documents - one that transcends document boundaries through the extraction of knowledge captured collectively within disparate document sentences. The names provide enhancements to the search results returned to users' queries, as described below.

4.1 Processing List-Type Queries

Let us consider a scenario involving a medical school student. Once she collects lab data, the user would like to find out how to process it automatically. Her next possible action is to submit a query such as *statistical packages* or *statistical package* to a Web search engine. The user has no previous experience with any such package. Therefore the set of the most representative software packages available constitutes a very desirable output from the search engine. In this and other scenarios, users search for lists of items by typing their category; other examples of list-type queries are *RISC processors* or *Clinton administration officials*.

To support list-type queries, when the input query matches a known category, the search results also include the top (i.e., most frequent) names as representative elements of that category. For instance, *SAS*, *SPSS*, *Minitab* and *BMDP* are returned in addition to the top documents for the query *statistical packages*. Similarly, *National Security Adviser Sandy Berger*, *Vice President Al Gore* and *Madeleine K. Albright* are returned for *Clinton administration officials*. Thus, categorized names supplement the regular search results for list-type queries. [5]

[3] A more relaxed insertion would insert a name under several WordNet concepts, based on a confidence threshold.

[4] More precisely, the evidence on the Web may be either truly non-existing, or simply unreachable through the extraction method presented in this paper.

[5] Visualization and user interface design issues, related to how exactly the names are displayed in the search results, are beyond the object of this paper.

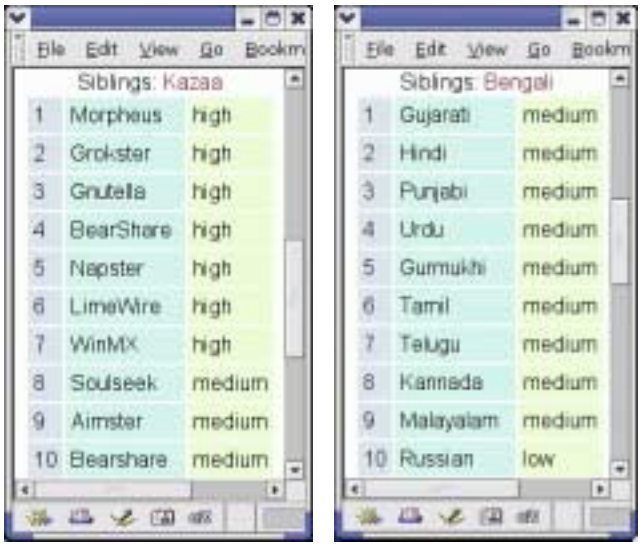

Figure 5: Top sibling names of *Kazaa* and *Bengali*

Table 3: Samples of siblings extracted from Web documents

Instance name	Top siblings
BMW M5	S-Type R, Audi S6, Porsche, Dodge Viper, Chevrolet Camaro, Ferrari
Information Retrieval	Natural Language Processing, Computer Vision, Spoken Language Processing, Digital Libraries, Human-Computer Interaction, Data Mining
NSA	CIA, FBI, INS, DIA, Navy, NASA, DEA, Secret Service, NIST, Army, DOE, NSF
Tangerine Dream	Kraftwerk, Klaus Schulze, Vangelis, Art of Noise, Jean-Michel Jarre, ELP, Genesis, Peter Hammill, Gentle Giant, Pink Floyd
Cozumel	Cancun, Belize, Jamaica, Puerto Vallarta, Bahamas, Mazatlan, Grand Cayman, Acapulco, Tulum, Isla Mujeres
Angels Landing	Narrows, Subway, Hidden Canyon, Emerald Pools, Weeping Rock, Watchman, Grotto Picnic Area, Great White Throne
CSCO	INTC, MSFT, SUNW, DELL, EMC, ORCL, AMAT, HET, WPI, WHR, TIN
Braque	Picasso, Matisse, Chagall, Leger, Dali, Kandinsky, Giacometti, Tamayo, Klee

4.2 Retrieval of Siblings

As mentioned earlier, a significant fraction of search engine logs are direct queries for names. Sometimes the queries refer to names that are completely unknown to the users, who want to find out about them. Concurrently, it is often the case, that the user is already familiar with other similar names. One who searches for *Kazaa* or *Bengali* may know already about the similar names *Napster* or *Hindi*. Retrieval of similar names, or sibling names, generally anchors the name into a set of possibly-known names, thus guiding the users in their search.

Figure 5 illustrates the top siblings retrieved from Web news articles for the names *Kazaa* and *Bengali*. The rank of a sibling is inversely proportional to its frequency of co-occurrence in the same category as the name being asked about. In the example, the frequency is converted to a threshold-based, three-valued reliability metric (*high*, *medium* and *low*). Note that siblings span across categories, e.g. *Tamil* is one of the *Indian languages* whereas *Russian* is one of the other *languages* besides *Bengali*. Table 3 shows other ranked siblings, which are extracted from Web documents rather than news articles.

From a user perspective, very small output sets might fail to anchor the input name. For instance, if the user is unaware what *Vodafone* is, and only one sibling is returned for it, namely *Orange*, the output has little use since *Orange* belongs simultaneously in *companies*, *counties*, *cities* etc. The addition of other, less-ambiguous siblings to the result set will provide the necessary disambiguation context. A similar problem occurs when the input name itself is ambiguous, e.g. the user searches for *Orange*. Since currently the siblings are retrieved across categories, the siblings of the most frequent category of that name are promoted to the detriment of the other siblings.

4.3 Query Refinement Suggestions

In the query refinement application, the goal is to suggest a few related queries, based on the analysis of the current query and its relationships with the acquired names and categories. Therefore query refinement suggestions have the potential of affecting both names and categories. If the query is a known name or a known category, a set of related queries is generated and offered to the user as suggestions for refinement. This is in contrast with list-type queries and retrieval of siblings, which operate on input that matches either categories or names.

If the input query is a known name, each related query consists of the name and one of its categories. Since a name may belong to many categories, a key issue is to decide which categories are the most representative. We use a ranking criterion that is complementary to the case of list-type queries, where the most representative names in a category are those names that are the most frequently found to belong to that category. For query refinement, the most representative categories for a name are those that are the most frequently found to contain that name. For illustration, according to the set of categorized names that we acquired in an experiment from Web documents, the 10 most representative categories for *Orange* are in order: *counties*, *operators*, *areas*, *companies*, *cities*, *topics* and *flavors*, *brands*, *organizations* and *colors*. Therefore the top 10 related queries suggested when the user types *"Orange"* would be *"Orange county"*, *"Orange operator"*, *"Orange area"*, *"Orange company"*, ..., *"Orange color"*.

When the input query is a known category (up to base form normalization), each related query consists of a category that is *RelatedTo* it. As discussed in Section 3, the concept of category relatedness builds upon the overlap of the instance sets. The overlap is measured by the cardinality of the set intersections relatively to the whole sets. The categories with the highest overlap are the most representative and therefore included in the related queries. The exception handles the high-overlap related categories that are more general based on simple lexical comparison. For illustration, this is the case for *companies* given *software companies*, or *cities* given *California cities*. Such more-general categories are deemed as useless and discarded from the set of query refinement suggestions.

Table 4: Test collections

Collection	Document type	Document count
NewsData	Web news articles	12 million
WebData	Web documents	≈500 million

5. EVALUATION AND RESULTS

5.1 Data

The experiments are performed on two document collections. They contain Web news articles (*NewsData*) and Web documents (*WebData*) respectively (see Table 4).

The news articles in *NewsData* are part of the Google News repository; they span from April 2003 until January 2004. The Web documents in *WebData* are a random selection from a snapshot of the Google index taken last year.

All documents are in English. Only the textual part of the documents is considered; everything else is ignored. Note that, on average, news articles are cleaner and more reliable than Web documents.

5.2 Results

Table 5 illustrates a few of the acquired categories and their instance names from Web documents. In the case of *hybrid cars*, the displayed set of instances is complete; for the others, only the top subset is shown. Instance names are ranked in decreasing order of their frequency within the given category. Thus, *Black*, *Los Angeles* and *Linux* have the highest frequency for *colors*, *California cities* and *operating systems* respectively.

Table 5: Samples of categories and their top ranked instance names acquired from *WebData*

Category	Top instance names
colors	Black, Red, White, Blue, Green, Yellow, Orange, Pink, Purple, Silver
hybrid cars	Toyota Prius, Honda Insight
California cities	Los Angeles, San Francisco, Sacramento, San Diego, San Jose, Oakland, Long Beach
rappers	Eminem, Jay-Z, Nas, Dmx, Snoop Dogg, Dr. Dre, Ja Rule, Mos Def, Nelly, Ice Cube
high-speed networks	ATM, Gigabit Ethernet, B-ISDN, FDDI, Myrinet, Frame Relay, Fast Ethernet
anti-depressants	Prozac, Zoloft, Paxil, Wellbutrin, Effexor, Elavil, Luvox, Nortriptyline, SSRIs
latin dances	Salsa, Merengue, Cha Cha, Rumba, Mambo, Tango, Samba, Meringue, Waltz, Cumbia
operating systems	Linux, Windows, Windows NT, Unix, DOS, Solaris, Microsoft Windows, MS-DOS, HP-UX

Intuitively, more general categories like *cities* capture larger sets of instance names than *California cities*. This was discussed and illustrated earlier in Figure 3. The results in Table 6 confirm the intuition on both *NewsData* and *WebData*. *Companies* are one of the important categories, followed by *players* in *NewsData* and preceded by *people* in *WebData*.

Table 7 shows a view opposite to Table 6, that is, instance names that belong to many categories, rather than categories with many instance names. In the *WebData* collection, the ubiquitous *Internet* is the instance name associated with the highest number of categories.

Among the top categories in Table 7, *clients* and *vendors* share a common property. Rather than (or in addition to) acting as regular class holders for their instance name sets,

Table 6: Categories containing the highest number of instance names

Noun Type	Collection	
	NewsData	*WebData*
Single	companies, players, organizations, areas	people, companies, names, organizations, artists
Multi	industry leaders, school districts, government agencies, brand names	professional organizations, industry leaders, government agencies, community organizations

they also encode "invisible" functional relations to other instance names. Indeed, a *vendor* is a vendor of something (e.g., a certain product). Similarly, a *client* could be a client of a given company. This corresponds to the notion of functions from a constant to another in first-order logic [24]. When abstracting away from unstructured text to categorized names, these kinds of functional relations are lost.

Table 7: Instance names that belong to the highest number of categories, and their top categories

Name	Top categories
From *NewsData*:	
China	countries, markets, nations, Asian countries, places, economies
IBM	companies, vendors, industry leaders manufacturers, OEMs, competitors
Japan	countries, nations, markets, Asian countries, places, trading partners
Microsoft	companies, vendors, industry leaders
France	countries, European countries, nations
Australia	countries, nations, markets, places
From *WebData*:	
Internet	technologies, networks, media, sources, services, resources, areas, electronic media
IBM	companies, vendors, corporations, industry leaders, manufacturers, clients, organizations
Microsoft	companies, vendors, industry leaders, clients, technology companies, corporations
Japan	countries, nations, markets, regions, Asian countries, places, foreign countries, areas
China	countries, nations, markets, Asian countries, places, regions, areas, economies
Oracle	companies, databases, vendors, industry leaders, relational databases, applications

Due to the diversity of the acquired categories, we do not currently have a complete qualitative evaluation in terms of precision and recall of all acquired names. Incipient evaluations indicate an average precision of 88%. The precision was computed over the top 50 instance names (or all names, if less than 50) of 20 randomly-selected, compound-noun categories. Examples of errors are *Singapore* categorized in *European capitals*, and *Dr. Dean Ornish* in *medical fields*. We plan to perform further evaluations of categories against other resources, e.g., gazetteers for categories that are of geographical nature.

The ranking induced by category size depends on the data source. For example, *California cities* are the 1184^{th} (*NewsData*) and 8220^{th} (*WebData*) largest category. Figure 6 illustrates the distribution of the largest categories according to the cardinality of their sets of unique instance names.

As the extraction method progresses through a new chunk of texts, some of the category-name pairs (C, N) that it en-

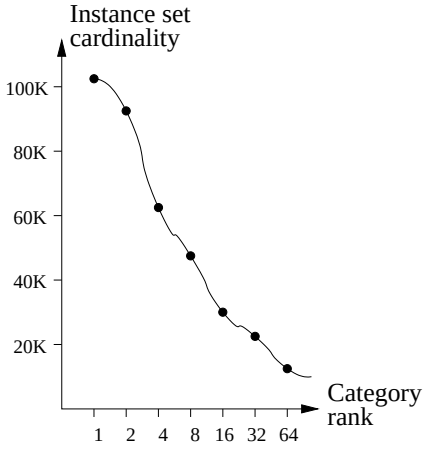

Figure 6: Frequency distribution of categories acquired from *WebData*

counters are duplicates of what was previously seen. Non-duplicates bring new information, whereas duplicates enforce previous information. In one experiment, the *NewsData* collection is split into chunks of documents based on the article posting date. Table 8 shows the percentage of unique items (pairs, categories or names) acquired from a new chunk of documents, that were not already acquired from previous chunks. For example, 16% of the unique pairs acquired from the fourth chunk were already seen in the previous chunks (1 through 3). The numbers are higher in the case of the names and categories that were already seen, with 39% and 35%. Over all *NewsData* chunks, Table 8 suggests that the percentage of unseen instances decreases asymptotically. However, it is unclear whether the process would saturate over a News collection that would be larger by several orders of magnitude.

Table 8: Incremental acquisition of categorized names from *NewsData*

Chunk#	Source (month)	Percentage of unseen unique items		
		Pairs (C,N)	Names only N	Categories only C
1	04/03	100%	100%	100%
2	05/03	91	76	76
3	06/03	70	69	69
4	07/03	84	61	65
5	08/03	82	60	61
6	09/03	82	57	59
7	10/03	83	53	58
8	11/03	80	54	56
9	12/03	79	52	54
10	01/04	79	51	54

6. PREVIOUS WORK

The importance of semantic lexicons and lists of names has been recognized in many text processing tasks such as prepositional attachment [3] and coreference resolution [18]. More importantly, many named entity recognizers traditionally rely on lists of names [16, 19]. The lists are compiled by humans, or assembled from authoritative sources. It is also possible to build recognizers that identify names automatically in text [7, 9, 26]. As opposed to the method presented in this paper, such approaches usually attempt to learn general categories such as *persons*, *organizations*, rather than refined categories. They also use a closed, pre-specified set of categories of interest, as a result of both explicit and implicit restrictions. In the first case, the training data introduces explicit restrictions. In the second case, it is the set of seed names, typically used in previous approaches, which introduces implicit restrictions on the acquired categories. Comparatively, we use an initial set of domain-independent patterns which leads to arbitrary categories.

In order to process natural language robustly, a variety of previous approaches apply lightweight techniques to unrestricted text [21, 22, 20]. In [1, 14] and other work, it is shown that patterns are useful for acquiring IsA and $InstanceOf$ information from unstructured text. Similarly, we focus on unstructured text as the source of our categorized names. Structural information such as HTML tags in Web documents offer a different means of extracting the same kind of relations [25]. The organization of acquired relations into larger structures [5], and the more ambitious objective of constructing knowledge bases from the Web, lead to projects like WebKB [8] and more recently [10].

7. CONCLUSIONS

The Web as a whole represents a huge repository of human knowledge. Most of the Web knowledge is not readily available in clean, structured, non-ambiguous form. Instead, it is encoded implicitly and scattered across billions of textual documents. This paper presented a lightly supervised method for accessing, decoding and exploiting a very small part of the information that Web texts wear on their sleeves. We showed that lightweight text processing techniques make it possible to acquire a very broad range of categorized instance names from unstructured text. The collected categories of names effectively fuse and summarize semantic relations detected within initially-isolated documents. In addition to enhancing existing knowledge resources, the acquired categorized names also enable novel Web search applications.

In our experiments, word capitalization was the only clue used to detect possible names in text. The simplicity of this heuristic cannot cover names containing numbers, for instance *7 Eleven* and *49ers*. In addition, the heuristic does not generalize to other languages where capitalization alone cannot distinguish between common and proper nouns, e.g. German. To increase precision and recall, we will explore other clues for finding candidate names, as well as a full-fledged iterative learning algorithm for detecting contextual extraction patterns.

Some of the semantics of each data-driven category is captured by its instance names. A direction to explore further is the recovery followed by discovery of relations among categories, based on the analysis of their sets of instance names. The relation recovery exploits existing resources to analyze the sets of instance names, e.g. for *countries* and *nations* which are known to be sometimes synonymous, or *African countries* and *countries* which are known to be in an IsA relation. The relation discovery then identifies relations among previously-unknown categories within unstructured text, e.g. a possible relation between *non-steroidal anti-inflammatory drugs* and *herbal medicines*. In a parallel effort, it will be useful to assess the strength of the relations, e.g. to what degree a category is $RelatedTo$ another.

The extraction patterns used in the paper focus on categorical facts. These facts usually capture the *genus* of the category to which an instance name belongs, e.g. *Prius* is an instance of *cars*. An equally useful piece of information would be the *differentia*. The differentia can be extracted from unstructured text with a different set of patterns, leading to descriptive rather than categorical facts. Examples of descriptive facts for *Prius* are *"uses a combination gasoline and electric engine"* and *"runs on electricity and petrol"*. Descriptive facts are a source of definitions when inserting names in existing knowledge resources like WordNet. In addition, the combination of descriptive and categorical facts is a step towards building large networks of interconnected categories, instance names and their associated facts.

8. ACKNOWLEDGMENTS

The author would like to thank Thorsten Brants for assistance with the TnT tagger; Vibhu Mittal and Jay Ponte for various suggestions; and Christoph Reichenbach, Mihai Surdeanu and Franz Och for feedback on earlier drafts.

9. REFERENCES

[1] E. Agichtein and L. Gravano. Snowball: Extracting relations from large plaintext collections. In *Proceedings of the 5th ACM International Conference on Digital Libraries (DL-00)*, San Antonio, Texas, 2000.

[2] T. Brants. TnT - a statistical part of speech tagger. In *Proceedings of the 6th Conference on Applied Natural Language Processing (ANLP-00)*, pages 224–231, Seattle, Washington, 2000.

[3] E. Brill and P. Resnik. A transformation-based approach to prepositional phrase attachment disambiguation. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, pages 1198–1204, Kyoto, Japan, 1994.

[4] S. Brin. Extracting patterns and relations from the World Wide Web. In *Proceedings of the 6th International Conference on Extending Database Technology (EDBT-98), Workshop on the Web and Databases*, pages 172–183, Valencia, Spain, 1998.

[5] S. Caraballo. Automatic construction of a hypernym-labeled noun hierarchy from text. In *Proceedings of the 37th International Conference on Computational Linguistics (ACL-99)*, pages 120–126, College Park, Maryland, 1999.

[6] N. Chinchor and E. Marsh. MUC-7 information extraction task definition, version 5.1. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, 1998.

[7] M. Collins and Y. Singer. Unsupervised models for named entity classification. In *Proceedings of the 1999 Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, pages 189–196, College Park, Maryland, 1999.

[8] M. Craven, D. DiPasquo, D. Freitag, A. McCallum, T. Mitchell, K. Nigam, and S. Slattery. Learning to construct knowledge bases from the World Wide Web. *Artificial Intelligence*, 118:69–113, 2000.

[9] S. Cucerzan and D. Yarowsky. Language independent named entity recognition combining morphological and contextual evidence. In *Proceedings of the 1999 Conference on Empirical Methods in Natural Language Processing and Very Large Corpora (EMNLP/VLC-99)*, pages 90–99, College Park, Maryland, 1999.

[10] O. Etzioni, M. Cafarella, D. Downey, S. Kok, A.-M. Popescu, T. Shaked, S. Soderland, D. Weld, and A. Yates. Web-scale information extraction in KnowItAll. In *Proceedings of the 13th World Wide Web Conference (WWW-04)*, New York, 2004.

[11] C. Fellbaum, editor. *WordNet: An Electronic Lexical Database and Some of its Applications*. MIT Press, 1998.

[12] S. Flank. A layered approach to nlp-based information retrieval. In *Proceedings of the 17th International Conference on Computational Linguistics and the 36th Annual Meeting of the Association for Computational Linguistics (COLING-ACL-98)*, pages 397–403, Montreal, Quebec, 1998.

[13] S. Green. Automatically generating hypertext in newspaper articles by computing semantic relatedness. In *Proceedings of the 2nd Conference on Computational Language Learning (CoNLL-98)*, pages 101–110, Sydney, Australia, 1998.

[14] M. Hearst. Automatic acquisition of hyponyms from large text corpora. In *Proceedings of the 14th International Conference on Computational Linguistics (COLING-92)*, pages 539–545, Nantes, France, 1992.

[15] B. Jansen. The effect of query complexity on Web searching results. *Information Research*, 6(1), October 2000.

[16] G. Krupka and K. Hausman. IsoQuest, Inc.: Description of the NetOwl extractor system as used for MUC-7. In *Proceedings of the 7th Message Understanding Conference (MUC-7)*, Fairfax, Virginia, 1998.

[17] M. Marcus, B. Santorini, and M. Marcinkiewicz. Building a large annotated corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2):313–330, June 1993.

[18] K. McCarthy and W. Lehnert. Using decision trees for coreference resolution. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence (IJCAI-95)*, pages 1050–1055, Montreal, Quebec, 1995.

[19] A. Mikheev, M. Moens, and C. Grover. Named entity recognition without gazetteers. In *Proceedings of the 10th Conference of the European Chapter of the Association for Computational Linguistics (EACL-99)*, pages 1–8, Bergen, Norway,, 1999.

[20] P. Pantel and D. Ravichandran. Automatically labeling semantic classes. In *Proceedings of the 2004 Human Language Technology Conference (HLT-NAACL-04)*, pages 321–328, Boston, Massachusetts, 2004.

[21] W. Phillips and E. Riloff. Exploiting strong syntactic heuristics and co-training to learn semantic lexicons. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP-02)*, pages 125–132, Philadelphia, Pennsylvania, 2002.

[22] D. Ravichandran and E. Hovy. Learning surface text patterns for a question answering system. In *Proceedings of the 40th Annual Meeting of the Association of Computational Linguistics (ACL-02)*, Philadelphia, Pennsylvania, 2002.

[23] E. Riloff and R. Jones. Learning dictionaries for information extraction by multi-level bootstrapping. In *Proceedings of the 16th National Conference on Artificial Intelligence (AAAI-99)*, pages 474–479, Orlando, Florida, 1999.

[24] S. Russell and P. Norvig. *Artificial Intelligence: a Modern Approach*. Prentice Hall, 2nd edition, 2003.

[25] K. Shinzato and K. Torisawa. Acquiring hyponymy relations from web documents. In *Proceedings of the 2004 Human Language Technology Conference (HLT-NAACL-04)*, pages 73–80, Boston, Massachusetts, 2004.

[26] M. Stevenson and R. Gaizauskas. Using corpus-derived name lists for named entity recognition. In *Proceedings of the 6th Conference on Applied Natural Language Processing (ANLP-00)*, Seattle, Washington, 2000.

[27] N. Stokes and J. Carthy. First story detection using a composite document representation. In *Proceedings of the 1st International Conference on Human Language Technology Research (HLT-01)*, San Diego, California, 2001.

[28] E. Voorhees. Using WordNet for text retrieval. In *WordNet, An Electronic Lexical Database*, pages 285–303. The MIT Press, 1998.