Robust Estimation of Google Counts for Social Network Extraction

ID: 103

Abstract

Various studies within NLP and Semantic Web use the so-called *Google count*, which is the hit count on a query returned by a search engine (not only Google). However, sometimes the Google count is unreliable, especially when the count is large, or when advanced operators such as `OR` and `NOT` are used. In this paper, we propose a novel algorithm that estimates the Google count robustly. It (i) uses the co-occurrence of terms as evidence to estimate the occurrence of a given word, and (ii) integrates multiple evidence for robust estimation. We evaluated our algorithm for more than 2000 queries on three datasets using Google, Yahoo! and MSN search engine. Our algorithm also provides estimate counts for any classifier that judges a web page as positive or negative. Consequently, we can estimate the number of documents with included references of a particular person (among namesakes) on the entire web.

Introduction

Recently, numerous studies of natural language processing (NLP) and the Semantic Web use a search engine for ontology construction, knowledge extraction, question answering, and other purposes. For example, Cimiano et al. develop a system called PANKOW, which enables automatic or semi-automatic annotation for the Semantic Web by investigating linguistic patterns of a named entity using Google (Cimiano, Handschuh, & Staab 2004). Matsuo et al. develop a social-network extraction system called POLYPHONET, which extracts social relations among persons using Google (2006). The bridge between a social network and a lightweight ontology is discussed by Mika (2005a; 2005b), where a social network and an ontology are extracted using a search engine.

The usage of search engines is also necessary for current and future NLP techniques, especially regarding the web as a large corpus (Kilgarriff 2003). The web is an excellent source of information about new words: Automatic thesaurus construction (Curran 2002) offers a great potential for various useful NLP applications. Several studies have examined the extraction of hypernyms and hyponyms or part-whole relations from the web (Miura, Tsuruoka, & Tsujii 2004; Cimiano, Handschuh, & Staab 2004; Hage, Kolb, & Schreiber 2006). Semantic similarity is calculated using a search engine and text processing (Turney 2005; Bollegala, Matsuo, & Isizuka 2007).

Most studies of the Semantic Web and NLP utilize a search engine, mainly for two purposes: to obtain a set of documents that include a given query, and to obtain statistical information, i.e. the count by a search engine called *Google count* of a given query. For example, if we obtain a large count from a query "Japanese food such as udon", we can automatically recognize that udon is an instance of Japanese food. Such patterns (popularly known as Hearst patterns (Hearst 1992)) are used to form an elaborate query, to recognize a hypernym and hyponym. Similarly, we can obtain the relevance of two terms. Some examples are that we can put "Yutaka Matsuo `AND` Peter Mika" to a search engine and recognize the strength of social relation between the two persons. Usually to improve the quality, the count is processed statistically. However, to make an additional improvement, we must devote greater attention to a search engine and carefully handle the Google counts.

Several widely known problems occur on Google counts: When the count is very large, it represents an approximation and not an exact value. Furthermore, because of the constant update of distributed indexes, the count might vary in a short time. The results obtained using Boolean operators such as `AND` and `OR` are sometimes erroneous. As a result, the counts sometimes violate basic set theory: It happens where the count of "A `AND` B" is more than the count of "A" [1]. Often the count of "A `AND` B" does not correspond to the count of "A" plus the count of "B" minus the count of "A `OR` B." Although such error-prone behavior of the Google count is rarely a concern for ordinary users, it does present difficulties for studies using Google counts for ontology extraction and social-network mining.

Therefore, it is important to make Google counts more precise for research purposes. In our approach, we do not rely on a single piece of evidence of a Google count. Rather we "estimate" the Google count based on multiple evidence obtained using a search engine so that the counts are not affected easily by a particular problem. The Google count is

[1] For example, "visualization" yields 36,800,000 hits, whereas "web `AND` visualization" makes 41,700,000 hits. As of November 18, 2006 by Google search engine.

estimated using multiple evidence of *co-occurrence counts*; the intuition is that if we want to have the count for query "A", we can apply query "A AND X" and obtain the co-occurrence count. The co-occurrence count is used to infer the original count of "A". The estimate would be robust if we were to use multiple co-occurrences as evidence.

In this paper, we use the term Google count because of the popularity of the term. We do not limit our scope of useful search engines to Google; more precisely, we might call it a search engine count. It is noteworthy that many search engines output exact hit counts. However, most commercial search engines for the entire web provide the approximate hit counts, which is partly attributable to distributed servers and indexing, and its frequent updates (Manning, Raghavan, & Schutze 2007). Therefore, our approach is beneficial to most search engines targeting the entire web.

This paper is organized as follows: we describe related works in the next section. Then, we explain our idea of Google-count estimation in detail. The algorithm is detailed, with subsequent evaluation using several datasets. We explain the application of our method to social network extraction, and conclude the paper.

Related Works

Few research efforts have directly examined the topic that we tackle in this study: but a couple of studies that investigate *sampling* using a search engine are relevant to ours.

Bar-Yossef and Gurevich propose a random sampling method from a search engine's index (2006). The method is applicable through a public search engine: the sampler performs a random walk on a virtual graph defined over the documents in the index, and thereby obtains near-uniform samples from the index. This method can be used, in principle, for robust estimation of Google counts: we can identify the occurrence of term A among sample documents and estimate its occurrence on the entire web. However, it requires many steps to obtain one estimate for query A.

A precedent work by Bharat and Broder (1998) presents a simpler method for sampling from a search engine's index. The algorithm formulates "random" queries generated from a lexicon of terms. The random queries are formulated based on the estimated frequency of each term in the lexicon. Although that method has some demerits for estimating the size and overlap of search engine indexes, it is appropriate for page-count estimation on the fly. If we apply the Bharat and Broder approach, we are able to estimate the page count for query A as follows:

- Make a lexicon L that consists of words and phrases. Obtain the prior probability $p(X)$ for each word/phrase $X \in L$.
- Put a query X ($X \in L$) to a search engine, and investigate whether the retrieved page contains word A or not.

This method requires vastly numerous samples to investigate the appearance of the given word A. If we are to measure the appearance of A by putting a query "A AND X", this approach is identical to our idea in this paper. Therefore, our approach is considered to be based on Bharat and Broder's work, and advancing it.

Method

Our approach starts from the following assumption: A Google count is precise when the number is smaller than k. Most search engines provide urls for the top k documents (e.g. k is 1000 for Google). If the count is less than k, the figure can be considered to be precise: technically we can check all k documents. In addition, a search engine usually provides all matched documents if they are appropriate. Especially if the hit count is small, search engines enumerate matched documents as much as possible. Therefore, the assumption should be reasonable.

Assuming that the Google count below k is precise, then the next step is to make the Google count of an arbitrary query to be less than k. By adding proper terms to the query, the Google count will be less than k eventually. More formally, assume that we want to have the Google count n_A for query A. Then, we add some query term X to A, and make query "X AND A," so that the count $n(X, A)$ would be below k. Throughout this paper, we denote the Google count of term A as n_A, and the Google count of term A with an auxiliary X as $n(X, A)$.

If X is distributed uniformly throughout all web documents, and if X and A are independent and identically distributed (iid), we can estimate the Google count of A. We denote the probability for a document to include word X as $p(X)$, then the estimate of the Google count is $n(X, A)/p(X)$. This is our basic procedure to estimate a Google count of query A using a more reliable count $n(X, A)$ and prior knowledge $p(X)$.

However, this approach is overly naive. Several crucial problems are readily apparent:

Estimation problem It is difficult to calculate the prior probability $p(X)$: it requires the correct number of all indexed documents and the number of documents including X. Both are difficult to obtain precisely.

Independence problem As we know through many NLP studies (Manning & Schütze 2002; Manning, Raghavan, & Schutze 2007), words show particular biases of co-occurrence with other words. It is not appropriate to assume that $p(X)$ is independent from query A.

Therefore, we take two solutions against the problems using the idea of a *magnification ratio* and selecting *probe words*, which we describe below.

Magnification Ratio Instead of assuming the prior probability $p(X)$, we learn the ratio of $n(X, A')$ to $n_{A'}$ on many examples $A' \in A_{train}$, where A_{train} is a set of words that we prepare for learning. In this way, we can know the ratio of $n_{A'}/n(X, A')$ empirically. The handling of the AND operator might have particular biases depending on a search engine. Therefore, this approach can be superior also from a theoretical point of view.

We call the average of $n_{A'}/n(X, A')$ on $A' \in A_{train}$ as *magnification ratio* of X on A_{train}, denoted as m_X. The estimation of the Google count n_A is calculated as $\hat{n}_A = m_X \times n(X, A)$. In a probabilistic sense, m_X corresponds to the average of $1/p(X)$ on a sample set A_{train}.

If a magnification ratio is large, it means the co-occurrence between the word and another word tends to be

small, thus the count would be more reliable. On the other hand, if the magnification ratio is too large, few documents will be retrieved, and the count would have less information. Therefore, it is important to use a word with a proper magnification ratio. In this study we do not use a word whose magnification ratio is below a threshold.

Probe Word To solve the independence problem, we carefully choose a word X. For example, the word *research* is a good word to estimate the Google count of a researcher's name because it is used broadly by many researchers. On the other hand, *mining* and *natural language* are less useful words for that purpose because they show an apparent bias of co-occurrences toward data mining researchers or NLP researchers.

Compare to many studies that use the co-occurrence bias in a corpus or a document, our approach is unique: most studies have used the biases of co-occurrence positively, e.g., for recognizing terms with co-occurrence biases as important (Matsuo & Ishizuka 2004). Our approach discovers words with less co-occurrence bias (or "unimportant words"). Usually, frequently appearing words are more representative in a document or a corpus, but in our approach, we do not want words with much appearance because they provide a small magnification ratio.

We call a carefully chosen word a *probe word*. Several criteria pertain to choosing a probe word, as we describe in the next section. For that procedure, we first prepare a set of words as candidates for probe words, denoted as X_{cand}. Then, we get a set of words X_{probe} so that $X \in X_{probe}$ shows less co-occurrence bias with words $A' \in A_{train}$.

Estimation from Multiple Evidence

Assume that we have a set of probe words. Our approach is based on each estimation by a probe word; the estimation is aggregated. Total estimation is executed using a certain function of the estimate counts; simply put, the average of the estimations is useful.

The Google count is considered as a Bernoulli distribution: We consider whether a web page matches to a query or not as a random variable. Under a proper transformation, a Bernoulli distribution can be approximated by a normal distribution. If we have m normal distributions corresponding to the m probe words, the mean can be estimated as $\mu = \frac{\sum_j \mu_j \times 1/\sigma_j^2}{\sum_j 1/\sigma_j^2}$, where μ_j and σ_j^2 respectively denote the mean and variance of distribution j. Therefore, we can make an estimation of the Google count of A as

$$\hat{n}_A = \frac{\sum_{X \in X_{probe}} m_X \times n(X, A) \times 1/\sigma_X^2}{\sum_{X \in X_{probe}} 1/\sigma_X^2}. \quad (1)$$

In other words, each estimate is weighted by the inverse of variance of the distribution. A probe word with large variance is less weighted.

Algorithm

The entire algorithm is illustrated in Fig. 1. We have two phases: the learning phase and the estimation phase. Below, we use a researcher domain as an example.

Learning phase:

1. Given A_{train} and X_{cand}.
2. For each $A' \in A_{train}$, obtain the Google count $n_{A'}$. For each $A' \in A_{train}$, obtain the co-occurrence to $X \in X_{cand}$, i.e. Google count $n(X, A')$.
3. Calculate the magnification ratio m_X, variance σ_X^2, and the bias-score s_X for $X \in X_{cand}$.
4. Among $X \in X_{cand}$, where m_X is above threshold m_{thre}, select a given number of $X \in X_{cand}$ with the lowest bias-score s_X . It is denoted as X_{probe}.

Estimate phase:

1. Given a query A.
2. Obtain the Google count $n(X, A)$ for each $X \in X_{probe}$.
3. Calculate

$$\hat{n}_A = \frac{\sum_{X \in X_{probe}} (m_X \times n(X, A)/\sigma_X^2)}{\sum_{X \in X_{probe}} (1/\sigma_X^2)}.$$

Figure 1: Algorithm for learning and estimation.

In the learning phase, a set of sample queries A_{train} is given, which we use for training. This set can be a lexicon consisting of named entities, general nouns, or personal names. We also prepare a set of candidate probe words X_{cand}. We obtain the page counts of $A' \in A_{train}$, and page counts of A' AND X where $A' \in A_{train}$ and $X \in X_{cand}$. We put an assumption that $n_{A'}$ is on average correct, i.e., the mean value of the distribution corresponds to the real value: We simply use the page count of A' as $n_{A'}$. Then, we calculate m_X (and variances) for each candidate probe word. Therefore, little human effort is needed for the learning phase, except preparing sample queries and candidate probe words.

In our example, A_{train} is a list of researcher names, and X_{cand} are research keywords that are extracted from the titles of research papers. The co-occurrence between a candidate word and a sample query is shown in Table 1. We can see that some words such as *information* and *system* provide numerous co-occurrences, but that other words such as *robot* and *generation* provide fewer co-occurrences. Some researchers, such as *Masaru Kitsuregawa* and *Hideaki Takeda*, have large counts. We can see that general words such as *information*, *sake*, and *system* have low variances, whereas specific words such as *robot* and *agent* have large variances. Some specific co-occurrence cases exist in the matrix: the count of *learning* AND *Yoichi Motomura* is 9320, which is far larger than counts for other people.

Selection of probe words can be accomplished in several measures: We use four bias-score functions to score probe word X, as follows:

Var variance of magnification ratio, i.e. $s_X = \sigma_X^2$

Chi We regard $p(A') = n_{A'}/(\sum_{A' \in A_{train}} n_{A'})$ as the unconditional probability and measure the biases on the observed co-occurrences $n(X, A')$ by χ^2 value: $s_X = \sum_{A' \in A_{train}} \frac{(n(X,A') - n_{X*} \cdot p(A'))^2}{n_{X*} \cdot p(A')}$, where $n_{X*} = \sum_{A' \in A_{train}} n(X, A')$.

Cos We calculate the cosine distance of $n_{A'}$ and $n(X, A')$

Table 1: A matrix for probe-word selection.

	single*	*system*	*support*	*information*	*sake*	*learning*	*robot*	*method*	*agent*	*generation*	*knowledge*
Toshihiko Kamishima	298	201	114	257	204	187	65	191	58	98	125
Shigeru Nakamaru	160	34	15	122	118	52	10	14	7	10	54
Daisuke Torii	103	71	51	78	74	48	14	43	48	27	37
Hideyuki Nakanishi	499	399	294	460	369	204	145	205	315	205	197
Seiji Koide	403	184	101	204	154	48	36	70	64	57	111
Hiroko Shoji	1510	582	658	1120	822	447	235	287	107	92	447
Yoichi Motomura	16300	11400	605	13900	10300	9320	664	595	303	349	614
Hideaki Takeda	23000	15100	11500	18800	13100	830	665	899	861	702	11700
Masaru Kitsuregawa	91100	86900	78600	90000	84000	856	479	9600	73500	659	674
...	...	...	...	...	...						
magnification ratio m_X		1.994	4.225	1.433	2.302	4.835	8.617	4.505	9.551	5.952	4.811
variance of m_X		3.792	26.978	0.684	5.804	32.90	104.7	29.88	126.2	48.44	30.25

All the words are translated from Japanese. The *single* means the Google count of $A' \in A_{train}$ without a probe word.

Table 2: A sorted list of candidate words by variance.

	word	magnify ratio	variance
1	*information*	1.432	0.684
2	*research*	1.741	2.532
3	*system*	1.993	3.793
4	*sake*	2.302	5.804
5	*technique*	2.856	8.980
...	...	...	...
184	*modeling*	16.44	350.8
185	*experience*	18.10	403.6
186	*tag*	18.08	408.9
187	*teaching material*	20.96	511.1
188	*handicapped*	21.86	636.7

for $A' \in A_{train}$. We define s_X as the inverse of the distance.

KL Kullbuck-Leibler information: it measures the difference between two probability distributions p and q: $s_X^{-1} = \sum_{A' \in A_{train}} p(A') \log \frac{p(A')}{q(A')}$, where $q(A') = n(X, A') / \sum_{A' \in A_{train}} n(X, A')$.

We pick up a given number of words whose bias-scores are smallest. We will compare the above four measures, in addition to random selection (**Rnd**), in the next section.

Table 2 shows a list of candidate words sorted by variance. We can see that some words are good in terms of variance, but not in terms of the magnification ratio. Because the words with a low magnification ratio are inappropriate, we set a threshold to the magnification ratio.

In the estimation phase, a query word A is input. Then, we make queries to a search engine to obtain co-occurrence counts $n(X, A)$. We calculate the weighted sum of each count, as shown in Eq. (1).

The algorithm requires a number of queries to a search engine: if we use m probe words and training data of l samples, we must make at least $m \times l$ queries in the learning phase. In the estimation phase, we require m queries to a search engine for a given query A. A tradeoff pertains between the number of probe words and the precision, but we show that 10–20 probe words are sufficient, as explained in the next section.

Evaluation

We prepare three datasets: *a social-network dataset (SN)*, *a general word dataset in Japanese (GWJ)*, and *a general word dataset in English (GWE)*.

SN dataset The social-network dataset consists of the data used in social-network extraction (Matsuo *et al.* 2006). We target the Japanese Society for Artificial Intelligence (JSAI). We use the names of 540 researchers and 188 keywords as candidate words X_{cand} that appeared frequently in the title and abstract of the papers.

GWJ dataset The dataset consists of general words in Japanese. We crawled category labels of Yahoo! Japan Directory. We use 413 labels on the top and the second levels of the hierarchy, and 200 words that frequently appeared in web pages in the categories. In literature on ontology learning from the web, a search engine directory is sometimes used as a target ontology. Thus we prepare such dataset for evaluation.

GWE dataset The dataset consists of general words in English, crawled from 500 category labels in Yahoo! Directory. In all, 200 words are used as candidate words that frequently appeared in web pages.

The evaluation is processed as follows: We make an estimate of the Google count for query $A \in A_{train}$ and calculate $\hat{n}_A$ based on the co-occurrences $n(X, A)$ to the probe words $X \in X_{probe}$. For this evaluation, we assume that the Google count n_A is correct. Then, the correlation of the estimation $\hat{n}_A$ and the correct value n_A is calculated: if the correlation is high, the algorithm can estimate the Google count based on co-occurrence counts. We use five-fold cross validation.

It is not always true that n_A is the correct value: Actually, our research is motivated by the belief that n_A is sometimes unreliable. The assumption here is that n_A is, on average, correct, i.e., the mean value of the distribution corresponds to the real value. Based on this assumption, we can compare performance among various methods.

Table 3 shows the result of correlation with the Google, Yahoo! and MSN search engine.[2] Looking at those results, Kullback-Leibler (KL) and chi-square (Chi) show good performance. Every method is better than the random selection (Rnd) of probe words. The SN and GWJ datasets (both in Japanese) produce correlations of about 0.8–0.9. The GWE dataset produces greater than 0.9 correlation, sometimes as high as 0.97. Therefore, we can reasonably esti-

[2]We set the number of probe words to be 20, and the threshold of magnification ratio 3.0.

Table 3: Correlation of estimation to correct counts using the three search engines.

Google

	Sgl+Rnd	Sgl+Var	Sgl+Cos	Sgl+Chi	Sgl+KL
SN	0.645	0.786	0.797	**0.850**	0.748
GWJ	0.653	0.827	0.734	0.862	0.822
GWE	0.846	**0.978**	0.797	0.944	0.827

Yahoo

	Sgl+Rnd	Sgl+Var	Sgl+Cos	Sgl+Chi	Sgl+KL
SN	0.635	0.650	0.802	**0.853**	0.770
GWJ	0.620	0.740	0.852	**0.868**	0.788
GWE	0.696	**0.969**	0.871	0.917	0.483

MSN

	Sgl+Rnd	Sgl+Var	Sgl+Cos	Sgl+Chi	Sgl+KL
SN	0.536	0.783	0.762	**0.850**	0.775
GWJ	0.416	0.794	0.820	0.816	**0.869**
GWE	0.554	0.961	0.488	0.929	**0.972**

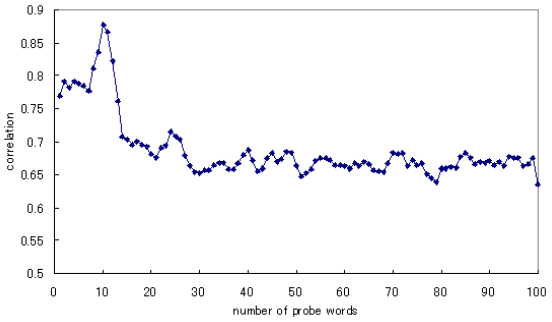

Figure 2: Number of probe words vs. correlation with KL scoring in the SN dataset.

mate the counts from co-occurrences. If we use KL or Chi for probe word selection, no differences exist in the performance among search engines. However, in other cases, especially when using Rnd, the Google result is better than the others. The number of probe words and performance are shown in Fig. 2: If more probe words are used (up to 12), the correlation increases. The performance is best if we use 12 probe words.

Figure 3 shows a scatter plot of estimated counts to correct counts for the SN dataset using the Google search engine. Although the estimates well correlate with the correct counts, we can see a strange gap in the distribution of correct counts. Namely, there are no queries that generate hit counts between 2000 and 9000. This phenomenon shows the existence of different modules that produce hit counts in Google search engine; when the number of retrieved documents is small, it provide the exact counts, and when many documents are retrieved, it provides approximation counts. We start our algorithm from the assumption; Google count is precise when the number is smaller than k. This phenomenon may validate the assumption.

Application to Social Network Extraction

An important expansion of our approach is to apply text-processing to the retrieved document and provide count estimates for the whole web. We build a *binary classifier*, which determines whether a given document is positive or negative.

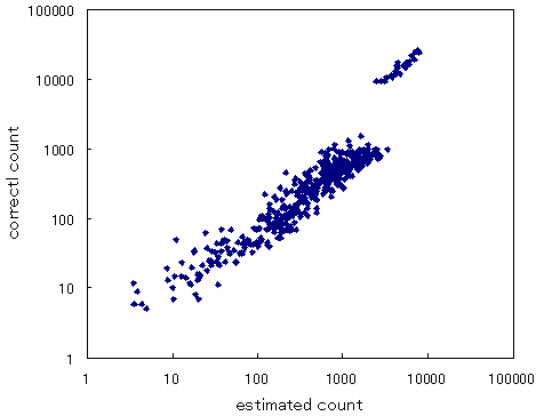

Figure 3: Scatter plot of estimated counts versus correct counts: We can see a gap of correct counts.

Table 4: Co-occurrence counts to the probe words with the binary classifier.

name	probe word	count	k_{pos}/k	modified count
Daisuke Torii	*support*	51	0.73	37.2
	structure	29	0.89	24.7
	behavior	36	0.75	27.0
	understand	29	0.90	26.1
Koichi Ando	*support*	166	0.82	136.1
	structure	145	0.76	110.2
	behavior	81	0.86	69.7
	understand	116	0.81	94.0
Manabu Okumura	*support*	902	0.91	820.8
	structure	945	0.76	718.2
	behavior	667	0.85	567.0
	understand	695	0.79	549.1

Table 5: Estimated counts of the persons.

name	without classifier	with classifier
Daisuke Torii	163.8	158.3
Koishi Ando	305.1	268.9
Manabu Okumura	17975.4	13967.2

Theoretically, any classifier is applicable that receives a web page and outputs positive or negative. For social-network mining, the classifier might judge whether the correct person (not a namesake) is mentioned in the document or not. Some might be interested in judging whether the word in the query is used in the intended meaning.

The problem here is how to estimate the positive counts of documents on the entire web. We put a query using the probe word. Then the positive count is multiplied by the magnification ratio. Assume that we have k documents and the classifier determines that k_{pos} documents are positive. Then the estimation of positive pages on the entire documents is $\hat{n}_A^{pos} = (k_{pos}/k)\hat{n}_A$.

We present some examples: If we want to know the Google counts of *Daisuke Torii*, *Koichi Ando* and *Manabu Okumura*, who are researchers of computer science in Japan. We obtain the co-occurrence count to probe words as shown in Table 4. Table 5 shows the modified estimates using an binary classifier. The classifier here outputs positive or negative values by judging whether or not the mention of a partic-

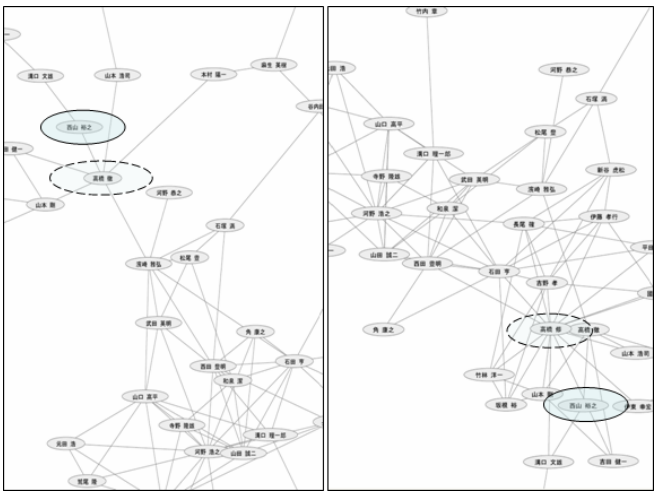

Figure 4: Two social networks using Google count (left) and using estimation (right): In both networks, the numbers of edges are the same (by tuning a threshold). Some nodes come to have numerous edges because the Google count is estimated lower.

ular person's name is about the target person (among many namesakes) (Bollegala, Matsuo, & Ishizuka 2006). In this example, in the case of *Daisuke Torii*, the estimated counts are similar whether we use the classifier or not. However, *Koichi Ando* produces about a 12% decrease of the count using the classifier, and in the case of *Manabu Okumura* the difference is about one quarter because *Koichi Ando* and *Manabu Okumura* are more common names than *Daisuke Torii*; consequently, the effect of the namesakes is greater.

As an example of application, we apply our method to social-network extraction (Matsuo *et al.* 2006). Figure 4 shows the two social networks of JSAI researchers, extracted from the web using either the naive Google count or estimation by our method. Because the Google counts for researchers with common names are sometimes larger than they should be, which causes less co-occurrence strength with other researcher names (by measuring Jaccard or Simpson coefficient), a small number of ties is recognized. However, if we apply our method, we can detect more ties for such researchers. For example, in the figure, we circled two persons who have fewer ties in the original network but more ties in our new network. Although the difference seems minor, it is important for more sophisticated algorithms to attain better quality of the extracted network.

Conclusion

In this paper, we proposed a novel algorithm that estimated the Google count robustly. The approach is based on the co-occurrence of words, as evidence to estimate the count of a given query. We described the effectiveness of our approach based on several datasets. Although that difference might seem minor, it is important to further advance many studies of the Semantic Web and NLP.

One contribution of this work is that it makes a binary classifier applicable to obtain Google counts on the entire web. Ontology extraction, social-network extraction and various NLP tasks can use Google counts with less concern for particular specifications and idiosyncratic search engine problems. Because the web offers its great potential as a corpus of social interaction and linguistic phenomena, we hope that our study will accelerate efforts toward various research using a search engine as an interface to the world's information.

References

Bar-Yossef, Z., and Gurevich, M. 2006. Random sampling from a search engine's index. In *Proc. WWW2006*.

Bharat, K., and Broder, A. 1998. A technique for measuring the relative size and overlap of public Web search engines. In *Proc. 7th WWW Conf.*

Bollegala, D.; Matsuo, Y.; and Ishizuka, M. 2006. Disambiguating personal names on the web using automatically extracted key phrases. In *Proc. ECAI 2006*.

Bollegala, D.; Matsuo, Y.; and Isizuka, M. 2007. Measuring semantic similarity between words using web search engines. In *Proc. WWW2007*.

Cimiano, P.; Handschuh, S.; and Staab, S. 2004. Towards the self-annotating web. In *Proc. WWW2004*, 462–471.

Curran, J. 2002. Ensemble methods for automatic thesaurus extraction. In *Proc. EMNLP 2002*.

Hage, W.; Kolb, H.; and Schreiber, G. 2006. A method for learning part-whole relations. In *Proc. ISWC2006*.

Hearst, M. 1992. Automatic acquisition of hyponyms from large text corpora. In *Proc. COLING'92*, 539–545.

Kilgarriff, A. 2003. Introduction to the special issue on the web as corpus. *Computer Linguistics* 29(3).

Manning, C., and Schütze, H. 2002. *Foundations of statistical natural language processing*. London: The MIT Press.

Manning, C.; Raghavan, P.; and Schutze, H. 2007. *Introduction to Information Retrieval*. Cambridge University Press. online version.

Matsuo, Y., and Ishizuka, M. 2004. Keyword extraction from a single document using word co-ocurrence statistical information. *International Journal on Artificial Intelligence Tools* 13(1):157–169.

Matsuo, Y.; Mori, J.; Hamasaki, M.; Takeda, H.; Nishimura, T.; Hasida, K.; and Ishizuka, M. 2006. POLYPHONET: An advanced social network extraction system. In *Proc. WWW 2006*.

Mika, P. 2005a. Flink: Semantic web technology for the extraction and analysis of social networks. *Journal of Web Semantics* 3(2).

Mika, P. 2005b. Ontologies are us: A unified model of social networks and semantics. In *Proc. ISWC2005*.

Miura, K.; Tsuruoka, Y.; and Tsujii, J. 2004. Automatic acquisition of concept relations from web documents with sense clustering. In *Proc. IJCNLP04*.

Turney, P. D. 2005. Measuring semantic similarity by latent relational analysis. In *Proc. IJCAI-05*.