Syntactic Complexity of Web Search Queries through the Lenses of Language Models, Networks and Users

Rishiraj Saha Roy[1]

Databases and Information Systems Group, Max Planck Institute for Informatics, Saarbrücken, Germany

Smith Agarwal[2]

Experian PLC, Cyberjaya, Malaysia

Niloy Ganguly

Department of Computer Science and Engineering, Indian Institute of Technology Kharagpur

Monojit Choudhury

Multilingual Systems Research Group, Microsoft Research India, Bangalore

Abstract

Across the world, millions of users interact with search engines every day to satisfy their information needs. As the Web grows bigger over time, such information needs, manifested through user search queries, also become more complex. However, there has been no systematic study that quantifies the structural complexity of Web search queries. In this research, we make an attempt towards understanding and characterizing the syntactic complexity of search queries using a multi-pronged approach. We use traditional statistical language modeling techniques to quantify and compare the perplexity of queries with natural language (NL). We then use complex network analysis for a comparative analysis of the topological properties of queries issued by real Web users and those generated by statistical models. Finally, we conduct experiments to study whether search engine users are able to identify real queries, when presented along with model-generated ones. The three complementary studies show that the syntactic structure of Web queries is more complex than what *n*-grams can capture, but simpler than NL. Queries, thus, seem to represent an intermediate stage between syntactic and non-syntactic communication.

Keywords: Query complexity, Statistical language models, Word co-occurrence networks, Crowdsourcing

1. Introduction

Searching information on the World Wide Web by issuing queries to commercial search engines is one of the most common activities engaged in by almost every Web user [1]. The Web has grown extensively over the past two decades, and search engines have kept pace by incorporating progressively smarter algorithms to keep all the information at our fingertips [2, 3, 4]. This co-evolution of the Web and search engines have driven users to formulate progressively longer and more complex queries, as seen by a rise in mean lengths from 2.4 through 3.5 to about four words per unique query over the last twelve years [5, 6, 7]. Search queries represent a unique mode of interaction between humans and artificial systems, and they differ observably in syntax from that of the parent NL. This has led researchers to argue that probably queries are acquiring linguistic properties of their own [8, 5, 9, 10, 7, 11]. Arguing from the perspective of the *function* of queries, i.e., communication, and the factors that influence their self-organization, it can be fairly convincingly established that queries are indeed an evolving *linguistic system* [7]. Nevertheless, there is no systematic and comprehensive study of the syntactic properties of Web queries that can convincingly bring out the fact that queries are indeed a "language". The challenge, of course, is to identify the unique syntactic features of an NL that make it different from any random or artificially generated sequence of symbols. Fortunately, there exist three lines of research that can address this fundamental question.

1.1. Background

One of the oldest statistical characterizations of NL comes from *n*-gram models that can be used for both generation of utterances or sentences to a certain degree of accuracy, and also

Email addresses: `rishiraj@mpi-inf.mpg.de` (Rishiraj Saha Roy), `smith.agarwal@my.experian.com` (Smith Agarwal), `niloy@cse.iitkgp.ernet.in` (Niloy Ganguly), `monojitc@microsoft.com` (Monojit Choudhury)

[1]This work was completed during the author's stay at the Indian Institute of Technology Kharagpur.

[2]This work was done during the author's internship at the Indian Institute of Technology Kharagpur.

for quantifying the predictability (and hence the complexity) of a system of symbols [12]. Such a line of research in the past has been extensively used for analyzing ancient languages (like the Indus script hypothesis [13]), studying languages with diverse typological properties [14], and also for understanding non-linguistic systems such as music [15] and the genetic code [16]. Over time, *n*-gram models have been appropriately generalized or restricted using more sophisticated linguistic features capturing various syntactic and semantic properties (see [17] for a review). Collectively, these models are studied under the broad topic of *statistical language modeling* and extensively used in applications like automatic speech recognition [18], machine translation [19], spelling correction [20] and information retrieval (IR) [21].

A second and more recent line of investigation into linguistic systems is through *complex network modeling* of languages, where a language is modeled as a network of *entities* and their *relations* (see [22] for a review). These studies were inspired by similar modeling techniques employed by physicists and biologists, which led to interesting insights into the systems being modeled. Such studies using network modeling have also revealed some interesting properties of languages [23, 24].

A third approach to characterize a linguistic interaction is to study it from the perspective of the *native speakers' intuition*, which says, to quote Noam Chomsky [25]: "The sentences generated will have to be acceptable to the native speaker". Though the concept of a *native speaker* is debatable and eludes a clear definition [26], in the context of queries it assumes an altogether new dimension, where it would refer to an average user of Web search engines.

1.2. Contributions

These three lines of investigation are, in fact, complementary, and therefore can be very well used for getting a more comprehensive picture of a linguistic system. The necessity of such a multi-pronged approach can also be appreciated in the context of a recent debate on the linguistic status of the Indus valley script: Based on the conditional entropy analysis of *n*-grams, Rao et al. [13] had claimed that the script was indeed used for an ancient linguistic system; this work was later fiercely criticized and the claim was contested by the computational linguistics community, led by Richard Sproat, who argued that even simple and random generative models can lead to such statistical properties [27]. Subsequent work on Indus scripts used network modeling to further substantiate Rao et al.'s original claim [28]. However, as we shall see, such an analysis, by itself, is not sufficient. Therefore, if queries are indeed an evolving linguistic system, then they should exhibit properties similar to NLs under statistical modeling, network modeling and cognitive analyses. Hence, in this work, we explore the syntactic properties of queries through these three different "lenses" and cross-validate our findings to come up with a holistic view. Specifically, we:

1. Build *n*-gram and *n*-term models [29, 30] from our Web query log and analyze perplexity (or predictability) of the models and compare them with that of Standard English;
2. Build word co-occurrence networks (WCNs), the most popular and well-studied network modeling approach for NLs, for the real log and compare the topological properties of the networks with those built from artificial logs generated using the *n*-gram and *n*-term models; and finally,
3. Ask ordinary Internet users to rate the acceptability or "real"-ness of the search queries generated by the various language models.

Our study reveals that although queries seem to be more predictable (or less complex) than NL, *n*-gram models still fall short of generating a rich set of artificial queries. A typical user is able to tell apart a real query from an artificially generated one, even though a trigram-based generative model seems to overfit the data and is capable of confusing the user. The word order in queries seems to be the most important clue helping a user to differentiate between the real and artificially generated queries. Hence, the structure of current Web search queries indicates a linguistic system that has at least a rudimentary word ordering constraint, and several other syntactic and semantic constraints that lie beyond the scope of *n*-gram and *n*-term models. In short, queries represent a system which is in between a fully syntactic and a non-syntactic communication system.

We do not know of any previous systematic study on these aspects of Web search queries, even though features like auto-complete and query suggestions are indirect evidence that predictability of query terms is indeed exploited by Web search engines. On the other hand, this study can not only help in more systematic and principled techniques in relevant applications, but can also have an impact on the way we view *query understanding* today. The complete code for this research has been made available at `https://github.com/iamrishiraj/QueryLMStudy` (accessed 02 February 2016).

1.3. Organization

The organization of the rest of this paper is as follows. First, we describe our dataset in Section 2. In Section 3, we present the generative language models for queries used in this work. Next, in Section 4, we discuss our complex network modeling technique and use the framework to compare the various generative models. We then present our experimental framework and results for crowdsourcing to measure human intuition of query syntax (Section 5). Subsequently, we discuss the implications of our results in Section 6, and make concluding remarks in the final section (Section 7).

2. Dataset

For all our experiments, we use a query log sampled from Bing Australia[3] in May 2010. This raw data slice consisted of 16.7M (M = Million) queries. We subsequently extracted 11.9M queries from the raw data such that the queries were composed of ASCII characters only and were of length between two and ten words. The justification for imposing a filter

[3]`http://www.bing.com/?cc=au`, Accessed 13 January 2016.

based on query length is as follows. One word queries do not contribute to network structures based on word *co-occurrence* and are not meaningful to be judged by humans (almost any word is potentially a search query). Very long queries (having more than ten words) are typically computer-generated messages or excerpts from NL text, and need separate query processing techniques (only 0.01% of our log). Out of the extracted queries, $4.7M$ are unique. In other words, the entire log contains $11.9M$ queries. However, there are duplicates in the log. If we remove the duplicates, we are left with $4.7M$ queries. But in order to preserve log properties arising out of the natural *power law* frequency distribution of queries [6], duplicates were retained for all experiments.

3. Statistical language modeling for queries

The ***n*-gram** language model (LM), which assumes that the probability of the n^{th} word in a sentence depends only on the previous $(n-1)$ words [12], has been one of the most popularly used generative language models for NL with many applications [31, 32, 19]. Therefore, as the first steps, we evaluate query models based on *n*-grams (and their variants). An *n*-gram, in our context, is any continuous sequence of *n* words from a query. Mathematically, an *n*-gram LM over sequences of words is defined by a Markov chain of order $(n-1)$ [33], and the probability of a *k*-word query is given by Equation 1:

$$P(w_1 w_2 \dots w_k) = \prod_{i=1 \dots k} P(w_i | w_{i-1} \dots w_{i-n+1}) \qquad (1)$$

Required probabilities can be initialized through training on the real query log from the relative *counts*, as shown in Equation 2 below:

$$P(w_i | w_{i-1} \dots w_{i-n+1}) = \frac{count(w_i \dots w_{i-n+1})}{count(w_{i-1} \dots w_{i-n+1})} \qquad (2)$$

Since queries are short (mean length was close to four words for distinct queries in our log), it is not practical to look beyond trigrams. Here distinct queries mean unique queries. Our query log, after removal of duplicate queries, contained $4.7M$ unique or distinct queries. The average length for these unique queries came out to be 3.98, very close to four words.

An ***n*-term** [29, 30] is an *unordered set* of *n* words, all of which occur in a query, but not necessarily next to each other. Since queries have been considered to be bags-of-words in several contexts [34, 35, 21, 36], a systematic exploration of *n*-terms becomes necessary. The meaning of an *n*-term is equivalent to an unordered skipgram [37]. We also note that *n*-term models are akin to the full dependence model [38], which can capture long-range dependencies between terms but ignore their ordering and proximity.

3.1. Query generative models

We explore five generative models for queries, viz. 1-gram, 2-gram, 3-gram, 2-term, and 3-term. Example queries generated by each of these models are shown in Table 1. 1-gram queries typically do not make much sense; they are just some random words thrown in to create a query, albeit maintaining the real word probability distribution. The notion of local coherence immediately becomes clear as one moves to 2-grams. The crossovers from one bigram to the next seem smooth, but as a whole the queries are generally not meaningful. Clearly, the chances of a query being meaningful in its entirety diminishes with increasing query length. The same thing can be observed for trigrams; but since several queries only have three, four or five words (can be generated with only one, two or three trigrams, respectively), such queries are often realistic. For 2-terms and 3-terms, we have similar observations but it is also apparent that the sequencing of the words is not very natural. Another point to note is that such models often contain word repetitions. Both these aspects are fall-outs of the relaxation of the strict ordering constraint.

Table 1: Three example queries generated by each of the five generative models. Queries, *n*-grams and *n*-terms are shown in `typewriter fontface` to differentiate them from running text throughout this paper.

Model	Example queries

1-gram	`a dhcp ephemeral detailing`
	`map rc2 western pacific kennedy`
	`anzac center catering civ integrate you guide`

2-gram	`create user account on roads`
	`access 2003 not working with info`
	`party poker table tennis player`

3-gram	`shaolin kung fu panda`
	`a brief history of witchcraft`
	`a thousand miles sheet music for websliders`

2-term	`acer 5310 for flash player`
	`adobe acrobat free download windows`
	`housing thailand what is cost of housing`

3-term	`adelaide entertainment explained seating plan area`
	`anti software virus windows vista`
	`adobe photoshop 7 for mac`

3.1.1. Query generation process

We now explain our process of generating artificial queries using *n*-grams or *n*-terms. We denote the query length (in words) as *L*, a random variable that can take integral values between two and ten. *Query length distribution* refers to the probability distribution of *L*, which is empirically estimated from the log. All the generation models described here are made to follow this distribution. The process for generating artificial queries using *n*-grams is as follows (the process is exactly analogous for *n*-terms): first a value *l* of *L* is stochastically sampled from the query length distribution. Then, an *n*-gram is sampled in proportion to its probability, forming the initial query. An extra word is added to this initial query trying to extend the previous string of $(n-1)$ words. This new word is chosen probabilistically from all the *n*-grams which have their first $(n-1)$ words as the string concerned. This process is continued till the desired query length, i.e., *l*, is reached. When a query generation process is unable to add a new word to a partially generated query using an *n*-gram model, then it *backs off* to an $(n-1)$-gram generative model to generate the new word.

Example of an *n*-gram model query generation. We explain the generation process with one of the example queries for 3-grams in Table 1, a thousand miles sheet music for websliders. First, we stochastically sample a query length seven from the real log query length distribution, which implies that the generated query will have seven words. Next, the 3-gram a thousand miles is sampled from the list of all real trigrams based on its occurrence probability. We now try to extend the query by looking at all 3-grams that start with the 2-gram thousand miles. Using similar frequency-biased sampling as above, we obtain the trigram thousand miles sheet. Next, a search for 3-grams beginning with miles sheet fails, and hence we *back off* to the 2-gram model and sample for 2-grams starting with sheet, which produces sheet music. We resume our search for trigrams using the rightmost 2-gram and add the words for and websliders in consecutive iterations. No backing off was required in the last two steps. Since we have now reached the desired query length of seven, the query generation process stops.

We do not choose a fixed length for all the queries in the artificial logs – we maintain the query length distribution of the real log. In other words, the proportion of queries of a particular length (in words) is approximately the same in an artificial log as in the real log. In our algorithm for generating artificial logs, we first stochastically pick a query length l and continue the query generation process till the query reaches l words, and then stop. Since our query generation process (for any language model) works by adding one word at a time, it is not difficult to stop the query at a particular number of words. For more discussion on this topic, see Section 3.1.2.

Example of an *n*-term model query generation. We explain the query generation process for an n-term model with one of the example queries in Table 1, adelaide entertainment explained seating plan area, using the 3-term model. First, we stochastically sample a query length six from the real log query length distribution, which implies that the generated query will have six words. Next, the 3-term {adelaide, explained, entertainment} gets sampled from the list of all real 3-terms based on its occurrence probability. Since the set of 3! = six 3-grams {adelaide explained entertainment, adelaide entertainment explained, explained adelaide entertainment, explained entertainment adelaide, entertainment adelaide explained, entertainment explained adelaide} are all indexed as one item {adelaide, entertainment, explained} in the 3-term probability list (individual words in a set arranged arbitrarily, not alphabetically), we arbitrarily select one of these to be the 3-gram in the query, say, adelaide entertainment explained. We now try to extend the query by selecting one out of all 3-terms that contain any two of the words in the 3-term {adelaide, entertainment, explained}. Now say such a search fails. We then *back off* to the 2-term model, which implies that we now look for a 2-term that contains only one of the words in {adelaide, entertainment, explained}. Using similar frequency-biased sampling as earlier, we now obtain the 2-term {adelaide, seating}. So the new word seating is appended to the query to obtain the partial query adelaide entertainment explained seating. Next, we resume the search for 3-terms containing any two of the words from the following set with four items: {adelaide, entertainment, explained, seating}. We sample {entertainment, seating, plan} and the new word plan is appended to the end of the query. In the next step, we sample {adelaide, seating, area} and so add area to the query. No backing off was required in the last two steps. Since we have now reached the desired query length of six, the query generation process stops.

We create the n-terms by the following process: we make one pass on the log and identify the unique n-terms (each query with k unique words contributes $C(k, n)$ unique n-terms, $k >= n$). In the same pass, we can also compute the frequencies of each of these n-terms, by incrementing an n-term count by one each time we encounter a duplicate. We then normalize the n-term frequencies with the number of queries in the log to obtain the n-term probabilities. Now, using these n-term probabilities, we follow the algorithm described earlier to generate the query. In short, we first stochastically sample a query length (in words) l from the real length distribution. Next, we choose an n-term in proportion to its occurrence probability. We then try to extend the query by one word by selecting an n-term which has $(n - 1)$ words (any $(n - 1)$ words, not necessarily the last) in common with the current (partial) query. This process is repeated using similar frequency-biased sampling of n-terms till the desired query length l is reached. In case an n-term is not found with the required words in common, we back off to an n-term model which is one order less than the current model.

Words are often repeated in n-term queries because of the following reason. When an n-term is selected for a query, it usually has a relatively high probability. Next, its words do not go out of contention for being reselected for the query. For example, if the 3-term {anti, software, virus} (anti software virus in the query) is selected, the next step is to look for a relatively high-probability 3-term with any two words of the query in common. It is quite probable that the 3-term {anti, virus, software} is selected again, with the query now becoming anti software virus anti or anti software virus software or anti software virus virus. In our model, we did not explicitly disallow the original n-term not to be reconsidered for query extension. Moreover, while extending the query in an n-term model, it is not only the last $(n - 1)$ words that are considered. Instead, the new n-term to be selected may have any $(n - 1)$ words of the query in common. For example, in the query housing thailand what is cost of housing, once we have housing thailand what is cost of, the next 3-term can have any of the two words from the set {housing, thailand, what, is, cost, of}. So, since a frequent n-term always has the possibility of being "reselected", we often see duplicate words in the n-term model-generated queries. The initial intuition behind our exploring the n-term models was to examine the hypothesis that queries were relatively free-word order, i.e., word order did not matter much for the user while formulating the query. We observed that n-term models failed to generate realistic queries, which gave us evidence that word order was indeed important (Section 6).

The algorithms for generating artificial logs from real logs are presented in Algorithms 1 (***n*-gram-Log-Gen**) and 2 (***n*-term-Log-Gen**). An *expression* refers to any *n*-gram or any *n*-term in the two algorithms, respectively. The backing off step is shown in bold in both algorithms. The steps unique to the *n*-term model are marked with comments in **boldface**.

3.1.2. *Query termination*

Deciding the final length of a query based on the partial query generated so far is an interesting possibility. However, it is non-trivial, or rather extremely hard to decide the length in the *n*-gram model based on the current partial query, using syntactic properties only. So we have taken a more natural and simpler alternative in the paper by using the query length distribution as a constraint. Further, similar works in NLP have used such an approach in the past – fixing the sentence length a priori [39, 40]. Moreover, deciding the current query length dynamically may make it very difficult to simulate the query length distribution in the real log. From past work in this area, we know that the subsequent co-occurrence network structure depends on the sentence length distribution [39, 40]. Thus, it would be difficult to have a fair comparison between real and model-generated query word networks if the length distributions do not match. Future research can focus on building a constrained generative model which would require a trade-off between achieving an expected length versus a more "natural" completion of the query, using syntactic observations only. In this context, we conducted the following experiment. Instead of words, we constrain the length by "segments", which are the individual syntactic units of queries and are possibly multi-word [41, 42, 43, 44]. Hence, they can be a better substitute for being the building blocks of queries instead of single words. Thus, it is worthwhile to explore how a generative model that uses length distributions based on segments would perform in this setup. So, we first segment our real log using the state-of-the-art query segmentation algorithm described in Saha Roy et al. [44]. We then compute the length distribution of the log in terms of segments and not words. We then use the 2-gram model to generate queries, but using segment 2-grams and segment occurrence probabilities instead of words. Follow-up experimental results on the segment-based generative model are discussed in Section 4.4.1.

3.2. *Measuring model perplexity*

Perplexity is one of the most common metrics used for evaluating *n*-gram systems [45]. It can be intuitively thought of as the *weighted average* number of values that a random variable can take. Thus, perplexity of an *n*-gram model tells us that if, on an average, a string of $(n-1)$ words of the language are known, how many words are likely to occur in the next position. This indicates the number of words that can follow a given $(n-1)$-gram in the language. A higher value means that the n^{th} word is less predictable from the previous $(n-1)$-word context. A higher perplexity value for a language model thus implies less certainty in the user's mind about its predictability. The perplexity of a probability distribution $p(x)$ of a random variable X is defined as $2^{H(X)}$, where $H(X)$ is the entropy of X and is given by Equation 3:

$$H(X) = -\sum_{x_i \in X} p(x_i) log_2 p(x_i) \tag{3}$$

Table 2: Perplexity and counts of the different models for NL and Web queries.

Model	NL (Perplexity)	Queries (Perplexity)	NL (Counts)	Queries (Counts)
1-gram	1,406.593	6,417.283	0.3M	0.2M
2-gram	193.722	104.337	3.5M	1M
3-gram	17.663	5.430	9.7M	1.1M
2-term	893.851	384.945	48.1M	4.2M
3-term	N.A.*	23.360	N.A.*	24.8M

* Dictionary runs out of memory even with 64 GB of RAM.

Computation method. For $n = 1$, the computations are straightforward, i.e. the probability distribution of unigrams is the one whose entropy (and thus perplexity) is calculated. For *n*-grams and *n*-terms where $n > 1$, the weighted average of entropies over all $(n-1)$-grams is considered as the entropy of the model. The weight is the probability of occurrence of the corresponding $(n-1)$-gram in the corpus. The entropy of a particular $(n-1)$-gram is the entropy of the probability distribution over all words that can appear in the n^{th} position given that the first $(n-1)$ words are fixed. This entropy, raised to the power of two, gives the corresponding perplexity. Specifically, every *n*-gram *slot*'s appearance is referred to by some random variable X, with probability distribution $p(x)$. The individual $p(x_i)$-s refer to the points in the probability distribution $p(x)$, or instantiations in which the slot can be filled. For an example with the 2-gram model, let X represent the 2-gram slot `<apple ...>`, where `...` can be filled by `juice`, `tart` or `pie`. Then, `apple juice`, `apple tart` and `apple pie` would be represented by the x_i-s (x_1, x_2, x_3), and their occurrence probabilities would be denoted by the individual $p(x_i)$-s $(p(x_1), p(x_2), p(x_3))$. Then, we would compute an entropy for `<apple ...>` using Equation 3 and the final entropy of the 2-gram model would be the weighted sum of the entropies of all such 2-gram slots, the individual weights being the occurrence probabilities of the respective 1-grams in the first positions of the 2-grams (like `apple`). The individual $p(x_i)$-s are < 1 and the individual $log_2(p(x_i))$-s are < 0, but the negative of the resultant summation, i.e., $H(X)$, is not less than zero. It varies between 0 and $log_2 N$, where N is the number of points in the probability distribution $p(x)$. Thus, the perplexity, which is $2^{H(X)}$, varies between 1 and N.

3.3. *Experimental results*

Table 2 reports the perplexities and counts of the different models for NL and Web search queries. The corresponding perplexity values can be obtained simply by raising 2 to the power of the entropy value $H(X)$ (as obtained from Equation 3). For NL, the corpus used contained 1M randomly sampled sentences

Algorithm 1 *n*-gram-Log-Gen(Q, n, $numQueries$)

```
Require: Real query log $Q$, model number $n$ ($n = 3$ for 3-gram model), no. of queries to be generated $numQueries$
$E \leftarrow \emptyset$    ▷ $E$ is the list of all expressions in $Q$ ($n$-grams, ($n$-1)-grams, ..., 1-grams), initialized to NULL
for all $q \in Q$ do    ▷ For each query $q$ in input log $Q$
    Extract all expressions $E_q$ from $q$    ▷ $n$-grams required for basic model, lower order expressions required for backoff
    for all expressions $e \in E_q$ do    ▷ For every expression $e$ in the query $q$
        if $e \notin E$ then    ▷ Expression $e$ not found in list $E$
            $E \leftarrow E \cup e$    ▷ Expression $e$ added to list $E$
            $count(e, Q) \leftarrow 1$    ▷ Frequency of $e$ in $Q$ is set to 1 when encountered for the first time in $Q$
        else    ▷ Expression $e$ has been encountered in $Q$ earlier
            $count(e, Q) \leftarrow count(e, Q) + 1$    ▷ Increment count of $e$ in $Q$ by one
        end if    ▷ end if $e \notin E$
    end for    ▷ end for all expressions $e \in E_q$
end for    ▷ end for all $q \in Q$
for all $e \in E$ do    ▷ For every expression $e$ in $Q$
    $probab(e, Q) \leftarrow count(e, Q)/|Q|$    ▷ Compute probabilities of expressions $e$ in $Q$
end for    ▷ end for all $e \in E$
Partition $E$ into $E^{(1)}, E^{(2)}, \ldots E^{(n)}$ by no. of words in expression    ▷ 1-grams go to $E^{(1)}$, 2-grams to $E^{(2)}, \ldots$, $n$-grams to $E^{(n)}$
$Q' \leftarrow \emptyset$    ▷ List of generated queries $Q'$ initialized to NULL
while $|Q'| < numQueries$ do    ▷ Generate queries till we reach the desired number $numQueries$
    $qlen \leftarrow$ stochastic-sample-fn($lendist$)    ▷ $lendist \sim$ Real-len-dist($Q$), we use a stochastic sampling function stochastic-sample-fn() to sample a query length $qlen$ from the word length distribution function of the real queries Real-len-dist()
    $q' \leftarrow$ “”    ▷ Query to be generated $q'$ initialized to empty string
    $e' \leftarrow$ freq-biased-sample-fn($E_n$)    ▷ We use a frequency-biased sample function freq-biased-sample-fn() to sample an expression $e'$ from a list in proportion to its occurrence probability in $Q$
    $q' \leftarrow concat(q', e')$    ▷ Concatenate $e'$ to $q'$
    while len($q'$) $< qlen$ do    ▷ Continue till we reach the desired query length
        $select\text{-}flag \leftarrow$ FALSE    ▷ $select\text{-}flag$ is a Boolean variable indicating whether we have found a new $e'$ to extend $q'$
        while $select\text{-}flag$ = FALSE do    ▷ $select\text{-}flag$ set to TRUE when new expression $e'$ is picked
            $x \leftarrow 1$    ▷ Try to select the next expression $e'$ for extending $q'$, $x$ acts as a model selector
            while $x \leq n - 1$ do    ▷ Continue till we reach the 1-gram model, $n - (n - 1) = 1$
                $E' \leftarrow \emptyset$    ▷ $E'$ will contain candidate list of expressions to choose from for query extension
                $q' \equiv q'[1\ 2\ \ldots\ k]$    ▷ $k$ words so far in $q'$
                if $x < n - 1$ then    ▷ We do not need to back off to the 1-gram model yet
                    for all $E_i^{(n-x+1)} \in E^{(n-x+1)}$ do    ▷ Superscript denotes appropriate partition by length of expression in words, subscript denotes individual expressions in set
                        if $q'[(k - (n - x))\ (k - (n - (x - 1)))\ \ldots\ (k - 1)\ k] \equiv E_i^{(n-x+1)}[1\ 2\ \ldots\ (n - x)]$ then    ▷ Last $(n - x)$ words of query so far $q'$ must be the same as the first $(n - x)$ words of an expression in $E^{(n-x+1)}$
                            $E' \leftarrow E' \cup E_i^{(n-x+1)}$    ▷ Add the satisfying expression to the candidate list
                        end if    ▷ end if $q'[(k - (n - x))\ (k - (n - (x - 1)))\ \ldots\ (k - 1)\ k] \equiv E_i^{(n-x+1)}[1\ 2\ \ldots\ (n - x)]$
                    end for    ▷ end for all $E_i^{(n-x+1)} \in E^{(n-x+1)}$
                else    ▷ We have reached the 1-gram model
                    $E' \leftarrow E^{(1)}$    ▷ Select a 1-gram
                end if    ▷ end if $x < n - 1$
                if $E' \neq \emptyset$ then    ▷ Non-zero candidate expressions found
                    break    ▷ Break loop
                else    ▷ No expressions found
                    $x \leftarrow x + 1$    ▷ Back off to lower order model
                end if    ▷ end if $E' \neq \emptyset$
            end while    ▷ end while $x \leq n - 1$
            $e' \leftarrow$ freq-biased-sample($E'$)    ▷ Sample from $E'$ in proportion to frequency
            $select\text{-}flag \leftarrow$ TRUE    ▷ An expression has been picked for query extension
        end while    ▷ end while $select\text{-}flag$ = FALSE
        $q' \leftarrow$ concat($q', e'[n - x + 1]$)    ▷ $e'$ has $(n - x + 1)$ words, we need to concatenate only the last word; we add one new word in every iteration
    end while    ▷ end while len($q'$) $< qlen$
    $Q' \leftarrow Q' \cup q'$    ▷ Add newly generated query $q'$ to output list $Q'$
end while    ▷ end while $|Q'| < numQueries$
return Q'    ▷ Return list of generated queries
```

from newswire data[4] in 2010. Newswire text was chosen because, in general, they contain cleaner NL sentences than random Web data. For comparability of NL values with queries, we kept the dataset size similar for the latter by randomly sampling $1M$ queries from our dataset. To preserve the natural frequency distribution, duplicates were not removed from either dataset. The NL text was case-folded and only alphanumeric characters (and whitespace) were retained. Perplexity values for Standard English reported in Brown et al. [46] are obtained from corresponding cross-entropy values and hence are not directly comparable to those in Table 2.

[4]`http://corpora.uni-leipzig.de/download.html`, Accessed 13 January 2016.

3.4. Interpretation

It is quite interesting to note that while the perplexity of the unigram model for queries is much higher than that of NLs, the perplexity of bigrams and trigrams show just the opposite trend. The explanation for this surprising trend is as follows. We observed in our data that the rate of encountering a new word for queries is much higher (about one per 20 words) than NL (about one per 58 words). Hence, the unigram distribution of queries is more diverse than NL. In fact, queries have a much larger specialized or *peripheral vocabulary* and very small *core vocabulary* as compared to NLs [23, 47]. This unique feature makes the perplexity of the unigram model very high for queries.

On the other hand, queries are also repeated and their repetition frequency is known to follow a power-law distribution [6]. In NL, sentences are rarely repeated exactly, except for phrases

Algorithm 2 n-term-Log-Gen(Q, n, $numQueries$)

```
Require: Real query log $Q$, model number $n$ ($n = 3$ for 3-term model), no. of queries to be generated $numQueries$
 1: $E \leftarrow \emptyset$    ▷ $E$ is the list of all expressions in $Q$ ($n$-terms, ($n$-1)-terms, ..., 1-terms), initialized to NULL
 2: for all $q \in Q$ do    ▷ For each query $q$ in input log $Q$
 3:     Extract all expressions $E_q$ from $q$    ▷ $n$-terms required for basic model, lower order expressions required for backoff
 4:     for all expressions $e \in E_q$ do    ▷ For every expression $e$ in the query $q$
 5:         if $e \notin E$ then    ▷ Expression $e$ not found in list $E$
 6:             $E \leftarrow E \cup e$    ▷ Expression $e$ added to list $E$
 7:             $count(e, Q) \leftarrow 1$    ▷ Frequency of $e$ in $Q$ is set to 1 when encountered for the first time in $Q$
 8:         else    ▷ Expression $e$ has been encountered in $Q$ earlier
 9:             $count(e, Q) \leftarrow count(e, Q) + 1$    ▷ Increment count of $e$ in $Q$ by one
10:         end if    ▷ end if $e \notin E$
11:     end for    ▷ end for all expressions $e \in E_q$
12: end for    ▷ end for all $q \in Q$
13: for all $e \in E$ do    ▷ For every expression $e$ in $Q$
14:     $probab(e, Q) \leftarrow count(e, Q)/|Q|$    ▷ Compute probabilities of expressions $e$ in $Q$
15: end for    ▷ end for all $e \in E$
16: Partition $E$ into $E^{(1)}, E^{(2)}, \ldots E^{(n)}$ by no. of words in expression    ▷ 1-terms go to $E^{(1)}$, 2-terms to $E^{(2)}, \ldots, n$-terms to $E^{(n)}$
17: $Q' \leftarrow \emptyset$    ▷ List of generated queries $Q'$ initialized to NULL
18: while $|Q'| < numQueries$ do    ▷ Generate queries till we reach the desired number $numQueries$
19:     $qlen \leftarrow$ stochastic-sample-fn($lendist$)    ▷ $lendist \sim$ Real-len-dist($Q$), we use a stochastic sampling function stochastic-sample-fn() to sample a query length $qlen$ from the word length distribution function of the real queries Real-len-dist()
20:     $q' \leftarrow$ “”    ▷ Query to be generated $q'$ initialized to empty string
21:     $e' \leftarrow$ freq-biased-sample-fn($E_n$)    ▷ We use a frequency-biased sample function freq-biased-sample-fn() to sample an expression $e'$ from a list in proportion to its occurrence probability in $Q$
22:     $e'_{gram} \leftarrow permute(e')$    ▷ **Create an $n$-gram $e'_{gram}$ by an arbitrary permutation of the words in $e'$**
23:     $q' \leftarrow concat(q', e'_{gram})$    ▷ **Concatenate $e'_{gram}$ to $q'$**
24:     while len($q'$) < $qlen$ do    ▷ Continue till we reach the desired query length
25:         $select\text{-}flag \leftarrow$ FALSE    ▷ $select\text{-}flag$ is a Boolean variable indicating whether we have found a new $e'$ to extend $q'$
26:         while $select\text{-}flag$ = FALSE do    ▷ $select\text{-}flag$ set to TRUE when new expression $e'$ is picked
27:             $x \leftarrow 1$    ▷ Try to select the next expression $e'$ for extending $q'$, $x$ acts as a model selector
28:             while $x \leq n - 1$ do    ▷ Continue till we reach the 1-term model, $n - (n - 1) = 1$
29:                 $E' \leftarrow \emptyset$    ▷ $E'$ will contain candidate list of expressions to choose from for query extension
30:                 if $x < n - 1$ then    ▷ We do not need to back off to the 1-term model yet
31:                     for all $E_i^{(n-x+1)} \in E^{(n-x+1)}$ do    ▷ Superscript denotes appropriate partition by length of expression in words, subscript denotes individual expressions in set
32:                         if $|set(q') \cap E_i^{(n-x+1)}| = n - x$ then    ▷ **Any $(n - x)$ words of query so far $q'$ must be the same as any $(n - x)$ words of an expression in $E^{(n-x+1)}$, the set() function converts a string into a bag-of-words set form, $n$-terms in $E^{(.)}$ are already in set form**
33:                             $E' \leftarrow E' \cup E_i^{(n-x+1)}$    ▷ Add the satisfying expression to the candidate list
34:                         end if    ▷ end if $|set(q') \cap E_i^{(n-x+1)}| = n - x$
35:                     end for    ▷ end for all $E_i^{(n-x+1)} \in E^{(n-x+1)}$
36:                 else    ▷ We have reached the 1-term model
37:                     $E' \leftarrow E^{(1)}$    ▷ Select a 1-term
38:                 end if    ▷ end if $x < n - 1$
39:                 if $E' \neq \emptyset$ then    ▷ Non-zero candidate expressions found
40:                     break    ▷ Break loop
41:                 else    ▷ No expressions found
42:                     $x \leftarrow x + 1$    ▷ **Back off to lower order model**
43:                 end if    ▷ end if $E' \neq \emptyset$
44:             end while    ▷ end while $x \leq n - 1$
45:             $e' \leftarrow$ freq-biased-sample($E'$)    ▷ Sample from $E'$ in proportion to frequency
46:             $select\text{-}flag \leftarrow$ TRUE    ▷ An expression has been picked for query extension
47:         end while    ▷ end while $select\text{-}flag$ = FALSE
48:         new-word $\leftarrow e' \setminus$ set($q'$)    ▷ **The new word to be added is the set difference between the words in the query so far $q'$ and the new expression $e'$**
49:         $q' \leftarrow$ concat($q'$, new-word)    ▷ **Concatenate the new word to the query so far**
50:     end while    ▷ end while len($q'$) < $qlen$
51:     $Q' \leftarrow Q' \cup q'$    ▷ Add newly generated query $q'$ to output list $Q'$
52: end while    ▷ end while $|Q'| < numQueries$
53: return Q'    ▷ Return list of generated queries
```

like `Thank you` and `Good morning`. In our dataset, the number of duplicates (repeated at least once) in NL and queries were found to be 447 and 164, 185 respectively. Furthermore, the mean sentence lengths for NL and queries were found to be 18.159 and 3.980 words, respectively. These two factors play a crucial role in bringing down the bigram and trigram perplexities for queries. One interesting conclusion from these findings on perplexity is that for a random sentence or query, a native speaker (or search engine user in case of queries) will be able to predict a random word present in an NL sentence much more certainly than for a query. On the other hand, if one or more words are shown, it is much easier to predict the rest of the words in a query than in an NL sentence. This is precisely why an autocomplete feature can work much better for Web search engines than for a word editor. An alternative perspective is as follows: Web search queries generated using bigrams or trigrams will look much more realistic than NL sentences generated using the same LMs.

The perplexities of n-terms are greater than their corresponding n-grams due to the manifold increase in the number of possibilities in the former (Table 2). The number of 3-grams is comparable to the number of the 2-grams for queries because of the presence of a substantial number of 2-word queries, that do not contribute to 3-gram counts.

The perplexity experiments were conducted to examine the hypothesis of queries evolving into a distinct linguistic system, different from the mother NL from which the words of the “query language” are borrowed. Had the n-gram model

Table 3: Acronyms used in this section.

Acronym	Expansion
LM	Language Model
WCN	Word Co-occurrence Network
LCC	Largest Connected Component
CC	Clustering Coefficient
ASPL	Average Shortest Path Length
KLD	Kullback-Leibler Divergence
DD	Degree Distribution
CDD	Cumulative Degree Distribution

perplexity values between NL and queries been almost similar, the above hypothesis could have been directly questioned. However, that was not the case and the values differed notably. Nevertheless, just having entropy or perplexity similar to or lower than NL need not, by itself, be indicative of an underlying language system [27], and hence further experimentation was deemed necessary. The low perplexity of trigram models in query logs can be related to the fact that trigram models are restrictive in generating synthetic queries and probably overfit the data.

4. Complex network modeling for queries

Network analysis provides an elegant mathematical framework to study various complex systems [48, 49, 50]. The success of such network-based techniques in the last couple of decades is primarily due to the fact that a network can capture aggregate properties of a system, while considering both local and long range (global) interactions present in a system. Of special interest to us here is the application of network models to linguistics and corpus studies [51, 22]. The most popular and well-studied representation of a language corpus is the *word co-occurrence network (WCN)* [23, 24, 52, 39], which we have applied here to study the statistical properties of query logs. In the recent past, such WCNs have also been used for term weighting in IR [53]. The acronyms used in this section are expanded in Table 3.

4.1. Network definition and construction

A WCN for any given text corpus is defined as a network $\mathcal{N} : \langle N, E\rangle$, where N is the set of nodes each labeled by a unique word and E is the set of edges. Co-occurrence can be defined on a query-level, sentence-level, paragraph-level, document-level, corpus-level or in a window of n words. In our model of a WCN, two nodes $\{i, j\} \in N$ are connected by an edge $(i, j) \in E$ if and only if i and j co-occur in a query (analogous to an NL sentence [23, 39]). Co-occurrence can be defined variously; in this paper, we define local and global models of co-occurrence as follows.

Local co-occurrence. According to this model of WCN, immediate word neighborhood is considered important and an edge is added between two words only if they occur within a distance of two (i.e. separated by zero or one word) in a query.

Global co-occurrence. In this model, an edge is added between two words if they occur within the same query, irrespective of their positions. Thus, a global co-occurrence network will have more edges than a local co-occurrence network.

For both local and global networks, the edges resulting from random and insignificant collocations are pruned using **edge restriction condition** [23] as follows. Let i and j be two distinct words from the corpus. Let p_i, p_j and p_{ij} be the probabilities of occurrence of i, j and the 2-gram $\langle i\ j\rangle$ (or 2-term $\{i, j\}$), respectively, in the data. Then, in a restricted network, an edge exists if and only if $p_{ij} > p_i p_j$. All networks considered in this study are undirected and unweighted. Figure 1 illustrates the concept of WCN by showing the network generated from the toy query log below. Pruned edges are shown using dashed lines.

```
samsung focus gprs config
dell laptop extreme gaming config
extreme gaming dell laptop config
buy samsung focus at&t
gprs config at&t samsung focus
samsung focus gprs config at&t
```

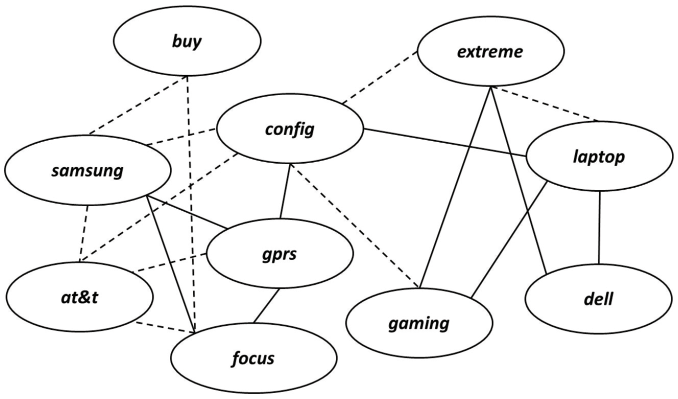


Figure 1: Illustration of a WCN for queries. It is built from the toy query log provided in the text. Pruned edges are shown using dashed lines.

As per convention [23], all network statistics discussed are computed on the largest connected component (LCC) of the graph. For LCCs of WCNs generated for $1M$ query samples from our query logs, $|N| \simeq 180,000$ (both for local and global models), while $|E| \simeq 1.5M$ (local) and $|E| \simeq 2.0M$ (global), on an average. Thus, these are very sparse networks with average edge density (i.e., probability of having an edge between a random pair of nodes = $|E|/\binom{|N|}{2}$) of the order of 10^{-4}.

4.2. Topological properties of WCNs

We now explain the topological properties of word co-occurrence networks that we use for characterizing real and generated query logs, namely, degree distribution, clustering coefficient, average shortest path length, and network motifs.

4.2.1. Degree distribution

The degree of a node in a network is the number of nodes that it is connected to. The degree distribution (DD) of a network is the probability distribution p_k of a node having a degree

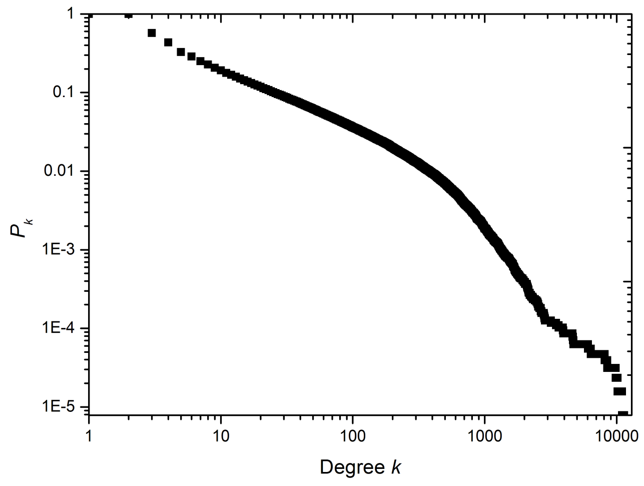


Figure 2: Sample Cumulative Degree Distribution (CDD) for a real query log. A CDD smoothens out the distribution, removing jitters in the DD graph by handling intermittent degrees with zero probability appropriately.

k. A cumulative degree distribution (CDD) P_k (probability of a node having degree $\geq k$) is more robust to noisy data points and is preferred for visualization. A representative CDD for a query WCN built from $1M$ randomly sampled queries is shown in Figure 2 (local model; global model is almost exactly similar) and is found to resemble a two-regime power law. This is also found for NLs [23] and indicates the presence of a *kernel-periphery* structure. A *kernel* is a small subgraph of the network where all nodes have very high degrees. The nodes in the *periphery* have relatively lower degrees than the nodes in the kernel. The majority of the nodes in a query WCN are observed to form very small peripheral clusters which are all connected to the kernel.

4.2.2. *Clustering coefficient*

Let a node r in the network have k neighbors. Then, $\binom{k}{2}$ edges are possible among its neighbors. The clustering coefficient (CC) of r, CC_r, is the fraction of these edges that actually exist in the network [54]. The $CC_{\mathcal{N}}$ of the entire network $\mathcal{N}$ is defined as the average of CC_r over all $r \in N$. A high CC indicates that the network consists of one or more dense subgraphs or clusters. The average CCs for the real WCNs (built from $1M$ random queries) are 0.429 (local) and 0.521 (global). The CC for the global network is slightly higher because of the higher edge density. These values are quite high as compared to the CC of an E-R random graph [55, 23], which is of the order of its edge density (10^{-4}). CC for similar networks for NL has been reported to be 0.437 [23]. This result shows lower density for NL WCNs than those for queries.

4.2.3. *Average shortest path length*

The shortest path length between a pair of nodes is the minimum number of edges that one must traverse to reach one node from the other. The average shortest path length (ASPL) is defined as the mean of the shortest path lengths between all pairs of nodes in the network that can be reached in a finite number of steps from each other. The ASPL for the WCN built from a $1M$ real query sample is 3.519 (local) and 3.004 (global). The ASPL for the global network is slightly lower because of the higher edge density. These values are quite small for a network with close to $180,000$ nodes, and near to the expected ASPL for an E-R random graph of similar size and density (4.253 (local) and 3.830 (global)). ASPL for an E-R random graph is given by $ln|N|/ln(\bar{k})$[54], where $\bar{k}$ is the average degree of the graph, given by $(2 \times |E|)/|N|$. It has been argued that the low ASPL for similarly constructed NL WCNs (2.67) is an outcome of an optimization of language structure necessary for fast recognition and retrieval of words [23]. As an aside, we also note that a network with high CC, low ASPL and low edge density (all with respect to random graphs) is known as a *small world network* [50]. Hence, like social networks and WCN for NLs, WCNs for queries are also examples of small world networks.

4.2.4. *Network motifs*

Network motifs are small subnetworks that are found to occur in significantly higher numbers in real networks than in random networks [56, 57, 58, 59, 60, 61]. For example, cliques with four nodes have an occurrence probability of 10^{-11} in a real WCN, while their expected probability in an E-R random graph [55] with the same number of nodes and edge density, is only 10^{-20}. A Ψ^n-motif is defined as a subgraph of n distinct nodes in the network, unique to structural isomorphism. Counting motifs for large graphs is computationally expensive, because beyond Ψ^4, motifs typically have a very large number of possible isomorphisms. In this study, we only consider connected Ψ^3 and Ψ^4 motifs. Figure 3 enumerates all the *connected* Ψ^3 and Ψ^4 motifs. The motifs in this paper are named following the convention introduced by Biemann et al. [39]. Motif detection has attracted attention as a useful technique to uncover structural design principles of networks in various domains like biochemistry, neurobiology, ecology, and engineering [57, 60].

Here, we use the algorithm FANMOD [60] to detect Ψ^3 and Ψ^4 motifs. Since motif counts are dependent on the size of a network, they must be suitably normalized by their corresponding expected counts for an E-R random graph model with the same number of nodes and edge density [60]. Since the ratios of the probabilities can be very skewed, we take the natural logarithms of these quantities for meaningful comparisons of values, which we shall refer to as the *Log Normalized Motif Counts* (LNMC), defined in Equation 4 below:

$$LNMC(\Psi_i^n) = log_e \frac{\text{Actual count of } \Psi_i^n}{\text{Expected count of } \Psi_i^n \text{ in an E-R graph}} \quad (4)$$

where, Ψ_i^n is the i^{th} n-sized motif. For example, following Figure 3, Ψ_2^3 and Ψ_3^4 would be the 3-clique and the 4-loop-out respectively. Figure 3 (third row) reports the LNMC values for Ψ^3 and Ψ^4 motifs for the real WCN (local co-occurrence model). This vector of eight elements for motifs is referred to as the *motif signature* of the network [39]. These ratios indicate that the probabilities of occurrence of all the connected motifs in a real WCN are several orders of magnitude higher than probabilities in an E-R random graph. Figure 3 also contains

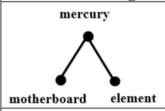
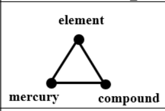
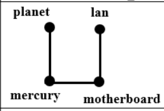
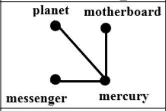
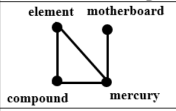
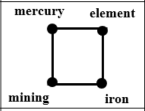
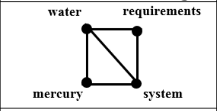
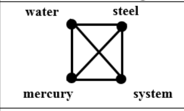
3-chain (ψ_1^3)	3-clique (ψ_2^3)	4-chain (ψ_1^4)	4-star (ψ_2^4)	4-loop-out (ψ_3^4)	4-box (ψ_4^4)	4-semi-clique (ψ_5^4)	4-clique (ψ_6^4)
							
2.941	7.185	4.101	6.057	10.315	7.212	15.463	21.484

Figure 3: Connected Ψ^3 and Ψ^4 motifs with examples and LNMCs from the real log. LNMC represents log normalized motif counts, i.e., logarithms of motif counts normalized by expected counts (or, corresponding probabilities) in an E-R random graph.

real examples of each type of motif from the network. Motifs in WCNs capture semantic relatedness between the words, and certain Ψ^4 motifs like *boxes* and *chains* are representative of semantic concepts like synonymy and polysemy, respectively [39]. We found motifs to offer insights into query semantics as well (Section 6).

Basic network properties like the numbers of nodes and edges, the degree distribution, the clustering coefficient and the average shortest path length are indeed very commonly used metrics for comparison. However, it is true that they cannot completely capture the complexity of the model generating the network, and can be replicated to a good extent using simple models. On the other hand, using motif signatures is a fairly new technique for this purpose, being proposed by Biemann et al. in 2012 [39]. While certain motifs like 4-cliques or 4-semi-cliques do represent strong connections among nodes in the network when present in substantially high counts, other motifs like chains, boxes, loop-outs and stars represent interesting semantic relationships among its nodes. Non-existent edges in the motifs represent non-connections between pairs of nodes, which may often convey non-trivial interpretations. Exactly replicating the motif signature of a network appears to be fairly hard, and thus the proximity of the motif signature of a network built from model-generated query logs (or NL sentences) to the signature of the network built from real logs can indeed shed light on the complexity of the model [39]. A motif signature is a more effective property for comparing networks at a macroscopic or aggregate level, than the standard characteristics like degree distribution, clustering coefficient and average shortest path length.

4.3. Stability of WCNs

The trends in the network statistics described above are useful only if they remain reasonably fixed when the size of the log is varied. Hence, to analyze the *stability* of the WCN statistics, we varied the network size by controlling the number of queries from which the network is created. Let $W (= 1M$ here) be the number of sample queries used to build a large WCN. We construct smaller query logs consisting of $W/10$, $W/20$, $W/50$, and $W/100$ queries by random subsampling of the entire log and computing the network statistics. To minimize sampling bias, for each W/s-sized log, the experiments were repeated s times and statistics were averaged over these s instances (not applicable for DD). Table 4 reports $|N|$, $|E|$, CC and ASPL for each of these sizes. Subscripts l and g represent local and global models of co-occurrence. Figure 4 shows the CDD plots for the respective networks (one specific sample from each network size; local co-occurrence model). In this figure, from the top left, we show degree distributions for networks constructed from query logs of sizes W, $W/10$, $W/20$, $W/50$, and $W/100$. We do not reduce the network size beyond $W/100$ because $W/1000 \simeq 1000$ queries which is too small for any reliable network analysis.

The results (mean values over s experiments) in Table 4 show that the statistics are extremely robust to network size variation. Similar stability for WCNs for NL text has been reported in in Biemann et al. [39]. Importantly, the standard deviations for the s experiments were found to be very low in all cases, further indicating network stability (of the orders of 10^{-3} for CC and 10^{-2} for ASPL respectively). Even when the dataset size is decreased by two orders of magnitude (W to $W/100$), the CC and ASPL change only by $\simeq 17\%$ (g) to 19% (l) and $\simeq 5\%$ (g) to 6% (l), respectively. We also note an interesting trend here – both CC and ASPL increase as the network size decreases. This is slightly counter-intuitive, because a large network with a small ASPL is expected to have a high CC. However, the trend is explained as follows. As the network grows in size due to an increase in the number of queries, (a) new nodes join the network with edges connecting them to existing nodes; and (b) new edges form between existing nodes. Since new nodes do not immediately have several new connections, the first event decreases the CC and increases the ASPL. The second event decreases the ASPL and increases the CC. However, these rates of increase and decrease of CC and ASPL caused by the two events are not the same. For Web queries, with lots of new and rare words (arising from various proper nouns) continuously joining the network, it is the first event that mostly dominates and is responsible for the drop in CC and increase in ASPL.

Examining the degree distribution, it is evident that the network stabilizes to its final form marked by a two-regime power law, at about $W/10$, which is $100,000 = 0.1M$ queries. The motif statistics for the local network indicate similar stabilization trends and have been shown in Table 5. Motif results for the global network also show similar behavior and hence are not reported here. From these results, we infer that for dependable query WCN analysis, one must have at least $0.1M$ queries in their sample.

Proximity of local and global co-occurrence networks. An important observation from these experiments on network stability is the relative invariance in the properties of local and global co-occurrence networks for queries. Even though the global networks have more edges, differences in the CC and ASPL values are very small. Moreover, trends observed in degree and motif distributions are also quite similar. Thus, in the rest of this paper, results will be reported only on local WCNs.

Table 4: Basic network statistics for various network sizes (local and global models indicated by subscripts l and g respectiely). The properties include the numbers of nodes and edges, the clustering coefficient and the average shortest path length, of the largest connected component in the network. W represents $1M$ queries.

Log size	$\|N\|_l$	$\|N\|_g$	$\|E\|_l$	$\|E\|_g$	CC_l	CC_g	$ASPL_l$	$ASPL_g$
W	177,900	177,975	1,526,043	2,089,729	0.429	0.521	3.519	3.320
$W/10$	46,528	46,523	299,496	412,218	0.463	0.555	3.536	3.328
$W/20$	30,398	30,399	169,670	233,932	0.474	0.566	3.567	3.355
$W/50$	17,160	17,162	77,052	106,306	0.493	0.588	3.644	3.410
$W/100$	10,991	10,990	41,270	56,992	0.512	0.611	3.741	3.481

Table 5: Ψ^3 and Ψ^4 Motif signatures at various network sizes (local co-occurrence). The vector composed of the LNMC values of the two 3-motifs and the six 4-motifs is referred to as the motif signature of a network.

Log size	3-chain	3-clique	4-chain	4-star	4-loop-out	4-box	4-semi-clique	4-clique
W	3.574	7.973	4.637	6.350	10.998	8.003	16.921	23.562
$W/10$	3.063	7.658	4.417	6.261	10.839	7.905	16.375	22.791
$W/20$	2.873	7.172	4.066	5.890	10.251	7.189	15.427	21.460
$W/50$	2.614	6.580	3.587	5.407	9.489	6.215	14.208	19.735
$W/100$	2.371	6.185	3.192	4.915	8.873	5.406	13.296	18.498

Table 6: Mean network statistics for the query LMs. Since DD cannot be summarized by a single average value, the Kullback-Leibler Divergence (KLD) values are computed between the DDs (after applying add-one Laplace smoothing) of the real networks and the DDs of the model-generated networks. The CC, ASPL and KLD values for the models with minimum deviations from those for the real network are shown in **boldface**.

Model	$\|N\|$	$\|E\|$	CC	ASPL	KLD
Real	34,209	242,680	0.623	3.302	0.000
1-gram	28,748	311,955	0.280	2.823	0.349
2-gram	33,257	209,947	**0.619**	5.011	0.077
3-gram	60,594	292,210	0.591	**3.472**	0.068
2-term	28,978	227,146	0.630	4.968	0.054
3-term	47,292	249,966	0.634	3.538	**0.031**

4.4. Comparison of real and model-generated query WCNs

We have seen that the statistical properties of the networks are stable as long as the log consists of at least $0.1M$ queries. Therefore, for reliable results, we decided to conduct all our experiments on logs having $1M$ queries. We sampled the entire real query log to construct 100 subsamples of $1M$ queries each, each subsample *preserving the query length distribution* by words. These will serve as our real logs for all the following experiments. Similarly, we stochastically generated 100 logs, each consisting of $1M$ queries, for each of the five generative models. We constructed the WCNs for these 6 (real and five models) $\times 100 = 600$ logs and computed the DD, CC, ASPL and motif signatures for each network. We observed negligible variance in the network statistics across the 100 samples generated from the same model, which further demonstrates the robustness of network modeling. Thus, we report only the average values for $|N|$, $|E|$, CC and ASPL in Table 6, and the average LNMC values for the connected Ψ^3 and Ψ^4 motifs in Table 7. We note that the values reported here for the real log do not necessarily match those corresponding to W in the previous sub-section because now the sampling is done preserving the query length distribution by words; the sampling was intentionally random during the experiments on network stability for emphasizing the idea of stability. Henceforth, we will refer to the WCNs generated from the real query logs as *real networks* and the WCNs generated from the model-based query logs as *model-generated networks*.

Since DD cannot be summarized by a single average value, the Kullback-Leibler Divergence (KLD) [62] is computed between the DDs (after applying add-one Laplace smoothing [63]) of the real networks and the DDs of the model-generated networks. These values are also reported in Table 6. The smaller the KLD, the closer is the DD of the network to that of the real WCN. Figure 5 shows the CDDs for one of the subsamples each from the real and the model-generated networks. The degree k is the number of nodes that a particular node is connected to, in the network. The plots have been split into two groups, to somewhat mitigate the problem of almost completely overlapping curves.

Smoothing for KLD. We smooth the graph DDs to make them suitable for KLD computation in the following way. The KLD between two probability distributions P and Q is defined as shown in Equation 5:

$$D_{KL}(P \parallel Q) = \sum_i P(i)\, ln \frac{P(i)}{Q(i)} \quad (5)$$

Finding the KLD requires that both the distributions sum to one. Hence, we use the simple (non-cumulative) DD. In all our experiments, P is taken to be the DD of the real graph R. Next, KLD is defined only if $Q(k) > 0$ for any k where $P(k) > 0$. If the quantity $0\ ln\ 0$ appears in the formula, it is interpreted as zero. Now, there will be many degrees k in a model-generated graph M such that no vertex has that degree. For such degrees, $Q(k) = 0$. But for some of the same degrees in R, we may have $P(k) \neq 0$, which is not permissible for KLD computation. To

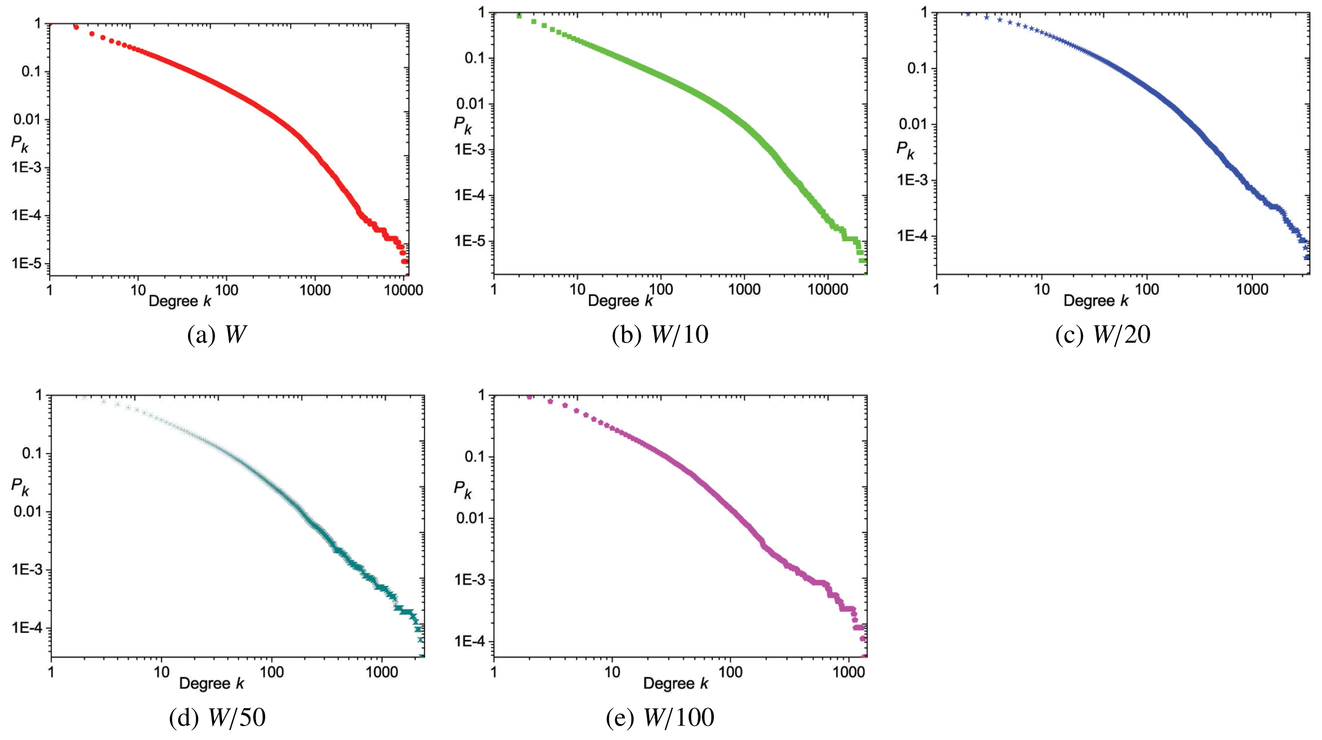


Figure 4: (Color online) Sample CDDs at different network sizes. Here W represents $1M$ queries. The network stabilizes to its final form marked by a two-regime power law, at about $W/10$.

avoid such cases, we must smooth the DD of M at all points where $Q(k)$ is zero. First, if the maximum degree for the real network ($k_{max}(R)$) is greater than that for M ($k_{max}(M)$), we set the maximum data point for Q to be at $k_{max}(R)$. Now for M, we first compute smoothed frequencies (number of nodes having a particular degree) and then compute the probabilities. Say, all the degree points $(a+i)$ in M between degree a and degree b are missing (zero), where i is an integer and $i_{min} = 1$, where $a < b$ and $freq(a) < freq(b)$. Then, we replace all these missing values with $freq(b) - freq(a)/(b - a)$, and this value is rounded off to the nearest integer. If it becomes zero, we add *one* to all frequencies in M (add-one Laplacian correction) [63]. We can similarly handle cases where $freq(a) > freq(b)$. If the last t degree points for M are missing (zero) and the last degree that has a non-zero frequency is m, then the frequencies of the last t points are replaced by $freq(m)/t$ and this value is rounded off to the nearest integer, and the add-one Laplacian correction is applied if necessary. Finally, the corresponding probabilities are computed by dividing the updated frequencies by the sum of the updated frequencies of all degrees of nodes in the network.

We can observe that even the DD of the 1-gram model is like a two-regime power law (Figure 5), which means that DD is the easiest of the network statistics to replicate. For other generated networks, the DD is almost identical to the real network, a fact also apparent from the KLD values in Table 6. However, the 1-gram model has much lower CC and ASPL than the real network. The CC matches (is close to the real network) for all models where $n \geq 2$, and the ASPL matches only for the 3-gram and 3-term LMs. Thus, ASPL is a harder statistic to replicate artificially than CC.

A general observation from the motif signatures is that like NLs [39], it is possible to be close to real networks on Ψ^3 motif counts with 2- and 3-grams. To examine deeper, we computed two aggregate motif statistics for each model: *M-Diff*, the motif-wise (pointwise) sum of the absolute LNMC differences between real and generated networks, and *M-Sum*, the sum of the LNMC values for all connected motifs. Equations 6 and 7 show computations of *M-Diff* and *M-Sum* for a model $\mathcal{M}$:

$$M\text{–}Diff(\mathcal{M}) = \sum_{k=3}^{4} \sum_{i} |LNMC(\Psi_i^k)_{Real} - LNMC(\Psi_i^k)_{LM}| \quad (6)$$

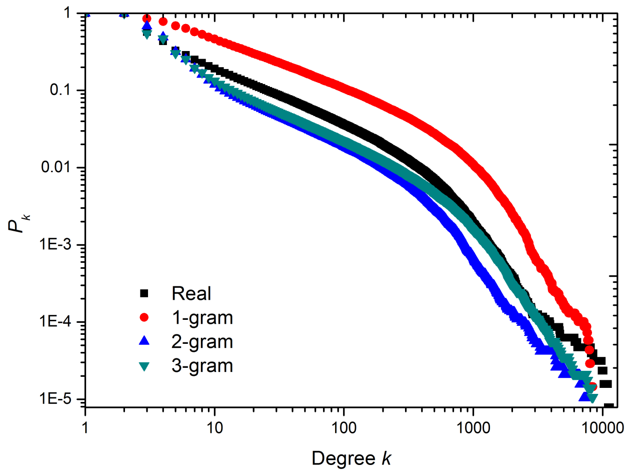
$$M\text{–}Sum(\mathcal{M}) = \sum_{k=3}^{4} \sum_{i} LNMC(\Psi_i^k)_{LM} \quad (7)$$

The existence of a large number of connected motifs in a network is an indication of its non-randomness [60]. We argue that the larger the sum of LNMC values for connected motifs, the more structured the network is. However, if this sum exceeds that of the real WCN, it implies that the network is becoming more restrictive than what is naturally expected. With *M-Diff* values as low as 1.590 and 1.835, the 2-gram and 2-term models have almost the same motif signatures as the real network. On the other hand, going by the abundance of connected motifs, the 3-gram and 3-term models seem to be the most restrictive (>

Table 7: Ψ^3 and Ψ^4 motif signatures for the query LMs. M-Diff represents the sum of the absolute differences between a model network's LNMC values with corresponding values from the real network. M-Sum is simply the sum of the individual values in a network's motif signature. The two minimum M-Diff-s and the two maximum M-Sum-s are shown in **boldface**.

Model	3-chain	3-clique	4-chain	4-star	4-loop-out	4-box	4-semi-clique	4-clique	M-Diff	M-Sum
Real	2.941	7.185	4.101	6.057	10.315	7.212	15.463	21.484	0.000	74.758
1-gram	2.579	6.428	3.706	4.868	8.987	6.627	13.710	19.072	8.781	65.977
2-gram	2.969	7.369	4.162	6.095	10.464	7.540	15.795	21.954	**1.590**	76.348
3-gram	2.971	8.049	4.631	6.001	11.383	8.286	17.281	24.153	8.109	**82.755**
2-term	2.874	7.073	3.989	5.733	9.935	7.210	15.075	21.034	**1.835**	72.923
3-term	2.907	7.832	4.439	5.859	11.057	7.967	16.832	23.514	6.113	**80.407**

The three lowest and the highest values in the M-Diff and M-Sum columns, respectively, are marked in **boldface**.



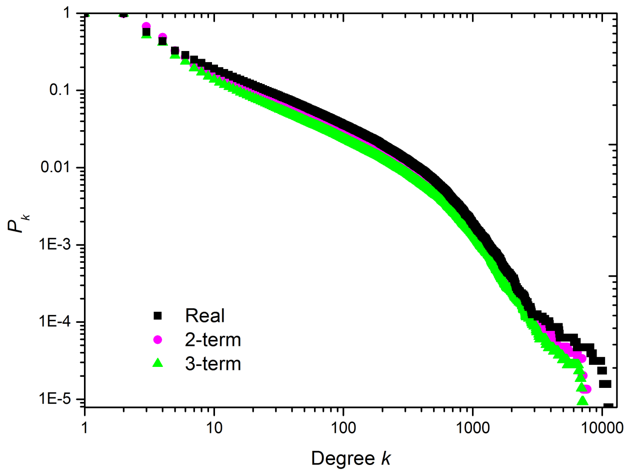


Figure 5: (Color online) Sample CDDs for query LMs. It shows CDDs for one of the subsamples each from the real and the model-generated networks. The plots have been split into two groups, to somewhat mitigate the problem of almost completely overlapping curves.

Log	$\|N\|$	$\|E\|$	CC	ASPL	KLD
Real	34,209	242,680	0.623	3.302	0.000
2-gram-*w*	28,748	311,955	0.280	2.823	0.349
2-gram-*seg*	270,687	1,642,641	0.572	3.519	0.523

Table 8: Basic network statistics for word (*w*) and segment (*seg*) co-occurrence networks for the same generative model 2-gram.

Real *M-Sum*). As supporting evidence, we note that the 3-gram model has lower perplexity than the 1-gram or 2-gram model (Table 2). We also note that the *n*-gram models have motif signatures closer to the real network than the corresponding *n*-term LMs. Thus, *relative word ordering in queries is important.* For example, in the query `convert pdf to word windows`, the intent of the query would be altered if the order of `pdf` and `word` was interchanged to `convert word to pdf windows`.

4.4.1. Segment co-occurrence networks

In continuation to the discussion in Section 3.1.2, we now evaluate how segment co-occurrence networks behave in the current framework. The basic network properties were computed on the LCCs and the results are presented in Table 8 and Figure 6. We observe that the segment 2-gram model (2-gram-*seg*) has matched up quite close to real logs (CC and ASPL); it is slightly nearer to real logs on ASPL and slightly farther away on CC and KLD, than the word-level 2-gram model 2-gram-*w*. The property values are quite close to real logs and to 2-gram-*w* in spite of the very large network size for segments than words (almost eight times larger both in terms of nodes and edges than the word level 2-gram-*w*). The number of nodes in the segment model are so high because segments are created by permutations of individual words, which results in a very large number of potential segments (although only a small fraction of the possible segments are actually present in the real query log). All values are averaged over 100 samples of 1*M* queries each. While it is intuitive that using segment statistics could ultimately produce better quality logs, the degree distribution shows a slightly higher deviation from the real distribution (the 2-gram-*seg* degree distribution plot is almost a one-regime instead of a two-regime power law) which needs a more thorough investigation. Segment-based query generative models remains an important avenue to explore.

4.5. Variation of the significance threshold

In our co-occurrence model, we consider a co-occurrence between a pair of words only if they occur together *more often*

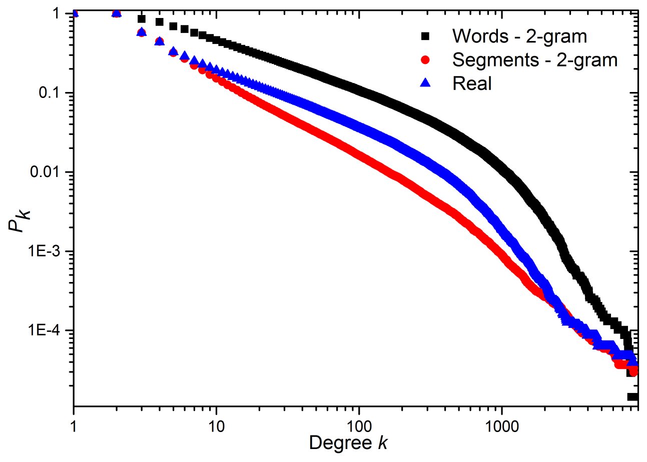


Figure 6: (Color online) Sample CDDs for word and segment co-occurrence networks for the same 2-gram generative model.

than expected by random chance. This is ensured by having an edge in the WCN only if the joint probability of occurrence p_{ij} of words i and j is greater than the product of their individual occurrence probabilities p_i and p_j, i.e. $p_{ij} > p_i \times p_j$ (edge restriction condition). This condition is imposed because if occurrences of i and j are independent, then $p_{ij} = p_i \times p_j$. Thus, edges are not created simply by a binary indicator, i.e., if the two words are co-occurring in queries or not.

However, weighted networks capture more information than unweighted graphs, and we have used the latter in our experimental model. Nevertheless, computing and comparing motif signatures between real and model-generated query logs are an integral part of our experimental framework, and counting motifs has not yet been defined for a weighted graph in the literature. However, we did perform a set of experiments on unweighted graphs that, in a way, simulates the effect of weighting the edges. We do this by having a parameter α as the exponent of the p_{ij} term in the edge restriction condition (Section 4.1), and varying α between 0.0 and 2.0 in steps of 0.1. Thus, our edge restriction condition becomes $p_{ij}^{\alpha} > p_i p_j$. When $\alpha = 0$, $p_{ij}^{\alpha} = 1$, and since $p_i \times p_j$ is always less 1 if probabilities p_i and p_j are both non-zero, we have a network with all the possible edges being present, irrespective of i and j co-occurring in a query. This is a boundary condition. As α is increased from 0.0 to 1.0 in small steps, the number of edges in the network decreases. Consider a specific pair of words i and j. Here a positive fraction is raised to the power of a positive fraction, and while the base of the quantity (p_{ij}) remains constant, the exponent (α) is increasing, and thus the value of the expression decreases. But the right-hand side of the condition remains constant ($p_i \times p_j$). Thus, it gradually becomes harder to have edges in the network. At $\alpha = 1.0$, we reach our original edge restriction condition. As α is gradually increased from 1.0 to 2.0, the value of left-hand side further decreases due to the nature of the values of the base and the exponent resulting in a continued decrease in the number of edges in the network. Thus, the number of the edges steadily decreases in the network as α is increased from 0.0 to 2.0. We found that at $\alpha = 2.0$, the network ceases to exist as none of the word pairs satisfy the edge restriction condition. *This framework with unweighted graphs has essentially the same effect on network properties as gradually increasing the weights of edges in a weighted-graph model.*

Table 9: Effect of variation of α in edge restriction condition on basic network properties of 2-gram model WCN. All computations are performed on the LCC of the network. Values are averaged over 100 samples.

α	$\|N\|$	$\|E\|$	CC	ASPL	KLD
0.0	33,213	225,370	0.656	3.739	0.084
0.1	33,213	225,370	0.656	3.739	0.084
0.2	33,213	225,370	0.656	3.739	0.084
0.3	33,213	225,370	0.656	3.739	0.084
0.4	33,213	225,370	0.656	3.739	0.084
0.5	33,213	225,370	0.656	3.739	0.084
0.6	33,213	225,370	0.656	3.739	0.084
0.7	33,213	225,366	0.656	3.739	0.085
0.8	33,213	225,229	0.655	3.741	0.084
0.9	33,213	222,777	0.644	3.739	0.083
1.0	**33,213**	**210,047**	**0.619**	**3.887**	**0.079**
1.1	33,213	182,645	0.595	4.137	0.081
1.2	33,210	147,534	0.579	4.466	0.091
1.3	32,962	112,668	0.569	4.942	0.128
1.4	32,564	82,731	0.562	5.865	0.202
1.5	30,828	57,438	0.562	7.807	0.329

The row corresponding to the original condition with $\alpha = 1.0$ is shown in boldface.

Specifically, we performed this α variation experiment on the logs generated by our 2-gram model. We report the basic network statistics and the motif signatures in Tables 9 and 10, respectively. As usual, all computations are performed on the LCC of the graph. The numbers reported are the average values of 100 random samples. There is negligible standard deviation among the properties of the random samples.

The number of edges falls sharply from $\alpha = 1.6$, with almost no edges being created from then on. The numbers of nodes in the LCC become 11, 6 and 0 for $\alpha = 1.6, 1.7, 1.8$ respectively, and the numbers of edges in the LCC become 17, 8 and 0, respectively. So we do not consider $\alpha > 1.5$ in our analysis.

We observe that with decreasing α, the number of edges decreases, making the network sparser. This is shown by the steady decrease of CC from 0.656 to 0.562. Similarly, as the edge density decreases, the shortest paths become longer, from 3.739 up to 7.807. We also observed that there is no change in the LCC from $\alpha = 0.0$ to 0.6, with the first noticeable decrease in edges in the LCC happening from $\alpha = 0.7$. The real network's CC and ASPL are 0.623 and 3.302 respectively. Since closeness to the real values is preferable, we find the best α to be 1.0 (CC) and <= 0.7 (ASPL). However, we see that there is no substantial deviation in CC or ASPL till about 1.0, after which the network starts becoming sparse at a reasonably high rate. The minimum KLD with the real degree distribution is again found to be the lowest for $\alpha = 1.0$, thus underlining the choice of α as 1.0 to be close to optimum (closeness to real logs is what we try to achieve). CDDs at some sample α values are shown in Figure 7. The plots are almost exactly overlapping

Table 10: Effect of variation of α in edge restriction condition on motif signatures of 2-gram model WCN. M-Diff decreases till $\alpha = 1.0$ and then increases. M-Sum decreases steadily till network breakdown.

α	3-chain	3-clique	4-chain	4-star	4-loop-out	4-box	4-semi-clique	4-clique	M-Diff	M-Sum
0.0	3.282	8.199	6.795	4.421	11.859	7.382	17.736	24.523	12.711	84.197
0.1	3.282	8.199	6.795	4.421	11.859	7.382	17.736	24.523	12.711	84.197
0.2	3.282	8.199	6.795	4.421	11.859	7.382	17.736	24.523	12.711	84.197
0.3	3.282	8.199	6.795	4.421	11.859	7.382	17.736	24.523	12.711	84.197
0.4	3.282	8.199	6.795	4.421	11.859	7.382	17.736	24.523	12.711	84.197
0.5	3.282	8.199	6.795	4.421	11.859	7.382	17.736	24.523	12.711	84.197
0.6	3.282	8.199	6.795	4.421	11.859	7.382	17.736	24.523	12.711	84.197
0.7	3.282	8.198	6.795	4.421	11.859	7.389	17.734	24.519	12.711	84.192
0.8	3.263	8.002	6.712	4.405	11.592	7.062	17.341	23.951	11.174	82.328
0.9	3.225	7.948	6.673	4.423	11.464	8.086	17.102	23.619	11.050	82.541
1.0	**2.969**	**7.369**	**4.162**	**6.095**	**10.464**	**7.540**	**15.795**	**21.954**	**1.590**	**76.348**
1.1	2.460	6.662	4.867	3.434	9.035	6.225	14.089	19.986	9.532	66.758
1.2	1.892	6.330	3.382	2.544	7.987	4.711	13.073	18.975	15.864	58.894
1.3	1.388	6.448	2.209	1.667	7.486	3.318	12.926	18.635	20.681	54.077
1.4	0.903	6.829	1.180	0.756	7.248	2.256	13.278	18.729	23.579	51.178
1.5	0.413	7.319	0.222	−0.146	7.148	1.504	14.009	18.902	25.655	49.370

The row corresponding to the original condition with $\alpha = 1.0$ is shown in boldface.

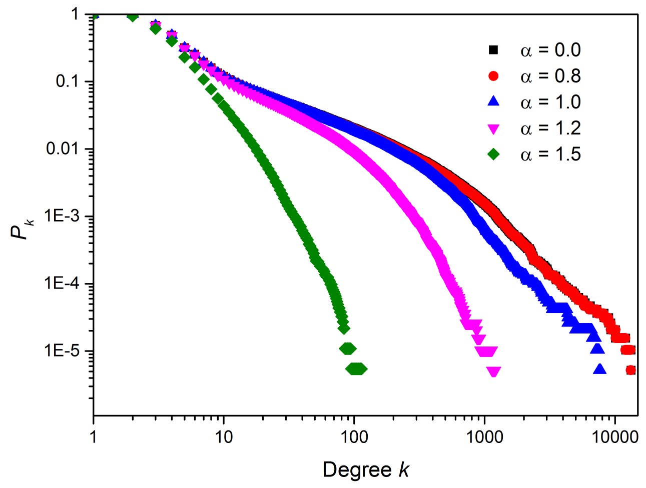


Figure 7: (Color online) Effect of variation of α in the edge restriction condition on CDD of 2-gram model WCN. The plots are almost exactly overlapping from $\alpha = 0.0$ to 1.0, and after $\alpha = 1.2$ the two-regime power law clearly breaks down, indicating a drastic change in network structure due to the rapid removal of edges.

from $\alpha = 0.0$ to 1.0, and after $\alpha = 1.2$ the two-regime power law clearly breaks down, indicating a drastic change in network structure due to the rapid removal of edges.

We observed that M-Diff initially decreased from 12.711 ($\alpha = 0.0$ through 0.7) to 5.329 (= 1.0, original). Then it increased steadily to 25.655 with increase in α. Since lower M-Diff is an indicator of proximity to the real network (Section 4.4), incidentally it turns out that $\alpha = 1.0$ is indeed the best setting. The M-Sum values, however, continue to decrease monotonically with increase in α from 0.0 to 1.5, throughout. However, as discussed within the text, the increase in the LNMC values only implies that the network is getting more restrictive and probably trying to overfit the training data. Thus, from these experiments, we find that with an exponent value of 1.0 for α, we get the closest proximity to the real network (the first row in Table 7).

Table 11: Effect of stopword removal on basic network properties of 2-gram model. Removing stopwords does not have a substantial effect on the basic network statistics of CC, ASPL and KLD from the real network.

Stopwords	$\|N\|$	$\|E\|$	CC	ASPL	KLD
Present	33,257	209,947	0.619	5.011	0.077
Absent	31,362	184,142	0.623	4.069	0.079

4.6. *Effect of stopword removal*

We wanted to study the effect of stopword removal. Again, we chose the 2-gram model as the representative for experimentation. We removed the stopwords from the logs generated by the 2-gram model and constructed the new WCN. We then compared the network statistics with those obtained from the original log containing the stopwords. Basic network properties are presented in Table 11 and the motif signatures in Table 12. Sample degree distributions are shown in Figure 8. From both the tables and the figure, we found that removing stopwords does not have a substantial effect on the network statistics. This is perhaps because users already omit most of the stopwords while formulating Web search queries.

5. User intuition of real queries

One of the important aspects of any NL is the grammatical correctness and coherence of the sentences, which is typically verified through native speakers' judgments [64, 65]. Native speakers can also predict the next word in a sentence

Table 12: Effect of stopword removal on motif signature of 2-gram model. Removing stopwords does not appear to substantially affect the motif signature.

Stopwords	3-chain	3-clique	4-chain	4-star	4-loop-out	4-box	4-semi-clique	4-clique
Present	2.969	7.369	4.162	6.095	10.464	7.540	15.795	21.954
Absent	2.567	7.374	4.920	4.016	10.212	7.370	15.770	22.072

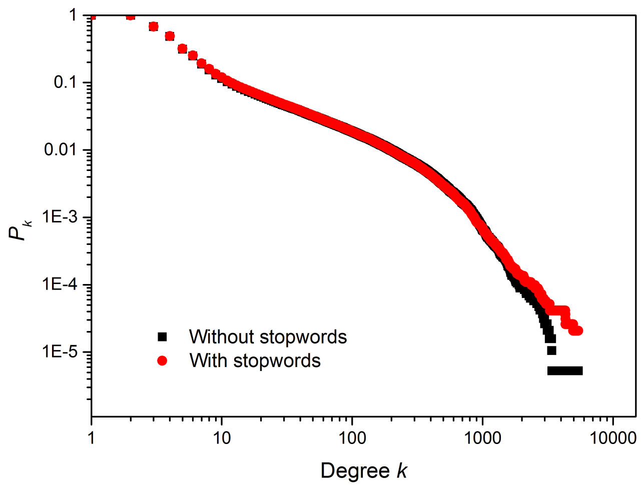


Figure 8: (Color online) Effect of stopword removal on the cumulative degree distribution of the 2-gram model. Removing stopwords does not seem to have a substantial effect on the cumulative degree distribution.

given the previous $(n - 1)$ words with a reasonable degree of accuracy [66], which makes them a good point of comparison against n-gram models. Therefore, statistical and network modeling-based analyses of query syntax would not be complete without a *native speaker* evaluation on acceptability of the generated queries. The challenge, however, is to redefine the concept of native speakers in the context of queries, and to design the corresponding query-acceptability task.

If queries are considered as a language, then clearly anybody generating a query can be considered a native speaker of the language. Thus, for our experiments, we deem an average search-engine user as the native speaker of the query language. However, asking a user whether a query is acceptable or not seems quite a meaningless task – any random sequence of keywords could constitute a query that has been issued by a real user, because as such there is no consensus on grammatical constraints on queries. To get around this problem we carefully designed our experiment in the following way:

1. Users were given a triplet having one real query and two generated queries.
2. They were asked to identify the real query in this triplet.
3. The remaining two search queries were to be rated on a five-point scale.

To make the comparison meaningful, the three queries in the triplet had some words (at least one word to at most three words) in common. The selection of query triplets was automated by a program; the program conditionally selected triplets from thousands of query sets if the three chosen queries of a triplet had at least one to at most three words in common. The philosophy behind this evaluation strategy is that if a generated query is sufficiently realistic, the user will have to make a random choice between the generated and the real query. Moreover, the rating points awarded to the queries in the triplet will give us information about the relative quality of the underlying generative models. We did not consider preference judgments [67] for the models as these are useful if one is only interested in the relative performance of the models. Here, on the other hand, we would like to find out the absolute goodness of a model with respect to real queries. This would not be captured through preference-based judgments. Moreover, after selecting the real query, since the users had to *score* the two remaining queries, the ranking within a triplet can be easily inferred.

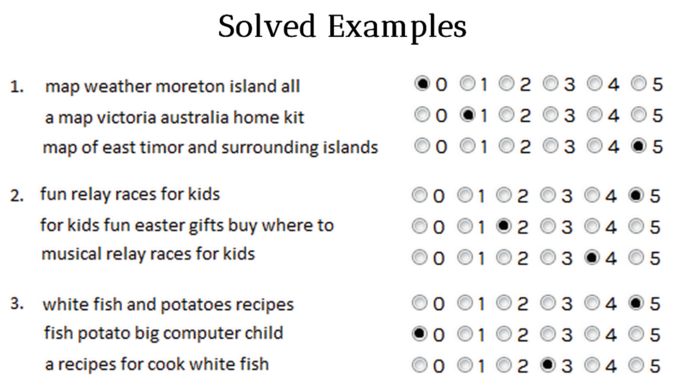
On a related note, Li et al. [68] describe an experiment where artificial queries were rejected if they were not acceptable to human judges. However, in their setup, new queries were created by string transformation methods from a real query. Users only had to judge if the generated queries had the same intent as the corresponding original ones, and therefore, their experimental framework is not applicable to our problem.

5.1. Method

We use crowdsourcing through Amazon's Mechanical Turk[5] (AMT) for our user experiments. Apart from being a cheap and fast method for gathering large data [69, 70, 71], a Turker (AMT task participant) is expected to be as good as an average search-engine user because AMT experiments are done online and often require one to really conduct Web searches. Hence, crowdsourcing is amenable to our experimental setup. We designed the Human Intelligence Task or HIT (a unit task in AMT) as follows. The five LMs can be combined with real queries to create $\binom{5}{2} = 10$ triplets. This gives us ten combinations of $\{$*Real query-Model i-Model j*$\}$ triplets. For each of these ten combinations, we randomly selected 35 triplets such that individual queries had some words in common (at least one word to at most three words) – making a total of $10 \times 35 = 350$ triplets containing $1,050$ queries to be judged. Some solved examples are shown in Figure 9. The choice of assigning a 5 appears difficult for question 2, and one of the realistic queries is model-generated. We assume that the annotator makes a random choice in such cases, and sufficiently realistic artificial queries will generally accumulate high scores in the end. To reduce annotator bias, we tightened our guidelines as far as possible, which are specified below:

1. Each question in the datasheet contains three queries – one is issued by a real human user, and the other two are generated by an algorithm.

[5]`https://www.mturk.com/mturk/welcome`, Accessed 13 January 2016.


Solved Examples

1. map weather moreton island all — [✓ 0] 1 2 3 4 5
   a map victoria australia home kit — 0 [✓ 1] 2 3 4 5
   map of east timor and surrounding islands — 0 1 2 3 4 [✓ 5]
2. fun relay races for kids — 0 1 2 3 4 [✓ 5]
   for kids fun easter gifts buy where to — 0 1 [✓ 2] 3 4 5
   musical relay races for kids — 0 1 2 3 [✓ 4] 5
3. white fish and potatoes recipes — 0 1 2 3 4 [✓ 5]
   fish potato big computer child — [✓ 0] 1 2 3 4 5
   a recipes for cook white fish — 0 1 2 [✓ 3] 4 5

Figure 9: Some solved examples for the crowdsourced query annotation task posted on AMT. To reduce annotator bias, we tightened our guidelines as far as possible, which are detailed in the text.

Table 13: AMT experiment details. An individual HIT was defined as the task of rating all the queries in five triplets. A set of five annotations was requested for each HIT.

Parameter	Details
Task description	Identify a real query hidden among model-generated ones. Then give grades to the remaining queries.
Task keywords	Web search queries, Real and generated queries, Rating queries
No. of query triplets in 1 HIT	5
Total no. of triplets	350
Annotations per triplet	5
Alloted assignment time for each HIT	20 minutes
Actual assignment time per HIT (mean)	1 minute 24 seconds
Turker qualification	Approval rate > 50 tasks
Turker location	Any
Reward per HIT	\$0.05
Total completion time allotted	7 days

2. The task is to find the query that is most likely to be issued by a human user and **mark it 5**. For example, `how to play a cd on my computer`.
3. The remaining queries are to be scored on a scale of 0 – 4 according to the following guidelines.
4. **Mark 4** if you think that the query is generated by an algorithm, but could almost be the real query (some external factor may make the query unrealistic, like the lack of the context in the example shown). For example, queries like `australian currency exchange limit`.
5. **Mark 3** if you think that the query is generated by an algorithm, and it makes sense, but has incorrect grammar or spelling[6]. For example, queries like `i fit to get blood transfussion`.
6. **Mark 2** if you think that the query is generated by an algorithm, and represents incomplete information needs or jumbled units, but could be meaningful if completed or reordered. For example, queries like `ancient rome slaves how did`.
7. **Mark 1** if you think that the query is generated by an algorithm, and parts of the query are coherent, but not as a whole. For example, queries like `creating pdf a file share tab security`.
8. **Mark 0** if you think that the query is generated by an algorithm, and it is totally nonsensical. For example, queries like `and anzac to jungle characters 101`.

Task details are presented in Table 13. An individual HIT was defined as the task of rating all the queries in *five triplets*. In other words, each HIT was defined by us as identifying the real query and scoring the other queries in the triplet for a set of five query triplets, i.e., a human annotator had to examine a set of five (independent) triplets and provide his annotations. In other words, (s)he may not choose to solve only three or four triplets in the task. Additionally, for a task to be approved by us, the annotator had to provide ratings to all the fifteen queries in the five triplets. Approval was subject to the condition of having exactly one score of five ("Real") in a triplet. We had requested and had obtained five annotations for every HIT. This means that we procured scorings by five different people for every query triplet (every query in every triplet). This was done to compute inter-annotator agreement (which turned out to be quite good) and allowed us to report aggregated ratings which are often more robust to noise than individual ratings. Aspects of the AMT experiment setup like cost, allowed time for each HIT and task descriptions are crucial to receiving quick and reliable responses. For our research, we followed the general guidelines presented in Alonso and Baeza-Yates [69].

Participants for our study needed to have at least fifty tasks approved in order to be eligible. Apart from this, no other restriction was placed on attributes of participants. Amazon allows any person to sign up for participation in AMT tasks, irrespective of his/her demographic properties. Consent of participants was obtained using Amazon. All relevant information can be found at Amazon's participation agreement[7] and the privacy notice[8]. None of the authors had any access to identifying information for the participants. AMT only provides user-ids to task publishers which lack any personally identifiable information.

5.2. *Results and observations*

Table 14 reports the average rating assigned to a query within a triplet by the five annotators, averaged over all queries from a model (reported in the "Average Rating" column). **#Total Triplets** count the number of triplets that has the presence of a query from the corresponding LM. The **"Real" Percentage** column lists the percentage of times a query generated by a model was marked as "Real". The actual number of times that a query from the corresponding LM was marked as "Real" in a triplet is shown in the **Judged "Real"** column.. The triplets

[6]In hindsight, this may not really have been a good criterion on which to judge the "real"-ness of a query. A spelling error can also be easily made by a human user.

[7]`https://www.mturk.com/mturk/conditionsofuse`, Accessed 13 January 2016.

[8]`https://www.mturk.com/mturk/privacynotice`, Accessed 13 January 2016.

Table 14: A summary of absolute ratings obtained through AMT. We report the average rating assigned to a query within a triplet by five annotators, averaged over all queries from a **Model** (reported in the **Average Rating** column). **#Total Triplets** count the number of triplets that has the presence of a query from the corresponding LM. The **"Real" Percentage** column lists the percentage of times a query generated by a model was marked as "Real". The actual number of times that a query from the corresponding LM was marked as "Real" in a triplet is shown in the **Judged "Real"** column.

Model	#Total Triplets	Judged "Real"	"Real" Percentage	Average Rating
Real	350	211	**60.317**	**4.046**
1-gram	140	14	9.645	2.406
2-gram	140	28	20.098	2.833
3-gram	140	39	**28.095**	**3.276**
2-term	140	25	18.072	2.880
3-term	140	22	15.625	2.875

The two highest values in the last two columns marked **bold**.

Table 15: Summary of relative ratings from AMT. Results are shown in a tournament-like fashion. Ordering of LMs in rows and columns is such that light and dark cells roughly create upper and lower triangular matrices.

Model	Real	3-gram	2-term	3-term	2-gram	1-gram
Real	*X*	0.655	0.741	0.752	0.708	0.832
3-gram	0.345	*X*	0.559	0.613	0.548	0.862
2-term	0.260	0.441	*X*	0.500	0.519	0.762
3-term	0.248	0.387	0.500	*X*	0.533	0.704
2-gram	0.292	0.452	0.482	0.467	*X*	0.423
1-gram	0.169	0.138	0.238	0.296	0.577	*X*

for which the annotation results obtained from AMT were inconsistent in any way (missing rating, and none or multiple 5-point ratings within a triplet) were re-published as new tasks till we obtained consistent ratings. We observed that real queries are detected correctly a large number of times ($\simeq$ 60%). It is notable that among generated queries, those from the 3-gram model were judged as "real" the greatest number of times and have the highest average rating of 3.276. Even though n-term models are poorer than corresponding n-gram models on being judged as real, their average ratings are marginally better than the 2-gram model.

Results are reported in a tournament-like fashion in Table 15. Ordering of LMs in rows and columns is such that light and dark cells roughly create upper and lower triangular matrices. The values are computed from all the triplets that had queries from both models i and j. Cell $[i][j]$ contains the fraction of times model i has won over (had a better rating than) model j. A dark cell indicates that the row lost substantially to the column (cell value < 0.4). A light cell indicates that the row and the column faired more or less equally well (cell value between 0.4 and 0.6). An unshaded cell indicates that the row won over the column substantially (cell value > 0.6). In this representation, the relative performance of the models becomes evident from the row (left to right, better to worse) or column orderings (top to bottom, better to worse). Not considering real queries which are identified correctly at least 65.5% of the times against the next best model (highest value in Row 1 of Table 15), all models have at least one grey cell in their rows (or columns). This indicates that even though the 3-gram model competes the best against real queries, it is not the best by a very big margin. Models in the middle zone are also quite comparable in their performance levels.

Inter-annotator agreement (IAA). In AMT, since a single Turker need not complete all annotations of the entire dataset, conventional ideas of IAA are not applicable. However, the average standard deviation for ratings from five annotators for a query is found to be 1.032. Given that the overall rating was to be done on a 6-point scale ($0-4$ and "Real" ratings), an average deviation of one point is within acceptable limits.

Effect of query lengths. We examined if the number of words in the generated queries had an effect on their perceived quality. We present mean user ratings for different query lengths in Table 16. We notice that the average ratings obtained for different query lengths are very similar, varying only from the second decimal place (between 2.721 for three words and 2.770 for five words). *Thus, with the generation process followed, the language model, and not the number of query words, play the determining role in how realistic the synthesized queries appear to Web users.*

5.3. Interpretation

These results provide us with the following interesting insights: (a) If any string was equally acceptable, real queries would get a "Real" rating only 33.3% of the time by random chance. The fact that real queries get the "Real" rating about 60% of the time implies that users already have a notion of queries being *well-formed*, i.e., the *acceptability* of queries; (b) The 3-gram model-generated queries can confuse the user about 28% of the time. In contrast, for NL, speakers can easily identify trigram-generated sentences, which are locally readable, but semantically incoherent [39]. This shows that trigrams capture more information than bigrams and probably overfit the data; (c) 2-term and 3-term queries getting lower "Real" percentage scores than 2-gram and 3-gram queries implies that word ordering provides vital clues to the users; (d) Bigrams received a higher "Real" percentage value but a lower average rating than the 2-term and the 3-term models. This is because bigrams generate very realistic queries at times, especially when the query length is small, but meaningless ones on other occasions. This is supported by the observation that the standard deviation of the ratings for bigrams is 1.476, while it is lower for 2-terms and 3-terms (1.334 and 1.302 respectively); (e) 2-term and 3-term models getting almost exactly similar average ratings further emphasizes the importance of word ordering. 3-terms are expected to generate semantically more coherent queries, but an obvious lack of ordering hinders their acceptability to Web users.

6. Discussion

We now discuss some of the inferences that we draw from the research presented in this paper.

Table 16: User ratings by query length. We notice that the average ratings obtained for different query lengths are very similar, varying only from the second decimal place (between 2.721 for three words and 2.770 for five words).

No. of query words	3	4	5	6	7	8
Average user rating	2.721	2.761	2.770	2.733	2.741	2.756

Motif analysis is insightful. Biemann et al. [39] observe that *box* motifs in NL (Figure 3) often occur due to *synonymy*, and chain motifs due to *polysemy*. We found that the synonymy and polysemy interpretations from NL hold good for queries as well. This is because synonyms will rarely co-occur in the same query, thus leading to an absence of connecting edges in the WCN. Similarly, the second and the third nodes in 4-chains (or the middle node in a 3-chain) are connected to concepts that are not connected to each other, hence representing multiple senses. Additionally, we observed that *box* motifs in query logs typically occur with two similar entities (e.g., `titanic` and `spiderman`, both movies) forming a pair of disconnected nodes and two attributes (e.g., `mp3` and `cast`) forming the other pair of disconnected nodes. The other common reason for box motifs in queries is *spelling mistakes or spelling variations* (like `pituitary` and `pitutiary`) at two opposite ends, and related words like `hormone` and `tumor` forming the other two opposite ends of the box motif. We observed *star* motifs in query logs arise due to a content unit (e.g., `titanic`) in the centre and three intent units [72] (e.g., `cast`, `mp3` and `review`) connected to it. Thus, motifs occur due to syntactic and distributional constraints in the linguistic system, and hence, they are relatively harder to capture through a generative model. While bigram models can match the motif signature of the real log to a good extent, the actual motifs are far less intuitive, which is also evident from the low acceptability of bigram-generated queries by Web users.

Trigrams overfit the data. User experiments show that trigram models generate quite realistic queries (Table 14) and have surprisingly low perplexity (Table 2). However, this does not imply that trigrams solve the problem of artificial query generation and thus can be used for creating synthetic logs. In fact, network analysis shows that trigram-generated queries give rise to a higher number of motifs of each kind than even the real WCN, which means that trigrams possibly overfit the training data. Therefore, a log generated with trigrams will contain realistic but frequently seen queries. Such a synthetic log will lack the diversity present in a true query log. This is not surprising because the average length of a query being only four, a large number of queries will be generated with only one or two trigrams. This will effectively generate only queries that have been frequently seen in the training query log.

There is scope for better generative models. Since trigram models overfit and bigram models fall short of generating good individual queries, realistic queries can only be generated using more sophisticated models that can capture the structural constraints of queries both at syntactic and semantic levels. Since motif analysis indicates the importance of content-intent relationships [72] in queries, we believe that a better quantification of the distribution of these relationships can lead to improved generative models for Web search queries.

Relative word ordering is important for the user. Researchers in the past have criticized the bag-of-words model for queries [73, 74, 75]. Our analysis strengthens earlier findings by showing the importance of word ordering constraints in queries, as the bag-of-words model-based query generation (using the *n*-term models) is shown to be inadequate in both network and user-based setups.

There is a cognitive model for queries. Finally, it is interesting to note that users, or as we can say, the native speakers of the language of queries, are indeed able to differentiate real queries from artificially generated ones. This shows that an average user has already internalized a cognitive model for queries, or for language in general. It is indeed believed to be true that there is a common faculty of language in all humans that helps in building the cognitive model for a language. Interestingly, this faculty mediating human communication appears very different from that of other living creatures; the human faculty of language bears resemblance to the genetic code – hierarchical, generative, recursive, and virtually limitless with respect to its scope of expression [76]. However, cognitive models built for the different languages will have different parameters that have to be internalized by its native "speakers". In this respect, we believe that the parameters for the language of queries (say, average length in words, relatively free word order, and omission of function words) are different from that of the parent language from which the query words are borrowed. Moreover, these parameters may depend on the parent language itself, i.e., "English" queries may have different parameters from "German" queries. Our experimental result that users are able to recognize real queries among synthesized ones about 60% of the time, shows that the average Web searcher has acquired the parameters of the English query language with a satisfactory level of proficiency. Further probing of this cognitive model through psycholinguistic experiments would be an interesting exercise that can provide interesting insights into the organization of query syntax, and also how a new language might evolve and automatically acquire a syntax of its own. It is worth noting here that since queries are not used for direct human communication, they cannot be collectively compared to the concept of a *register* in linguistics [77], which is a style of language used for a specific reason or in a specific social scenario. Finally, we do not know of prior work that takes a holistic approach towards the analysis of the syntactic complexity of Web search queries from the first principles. Previous works related to statistical and network analysis of queries have been reported in the respective sections. Nevertheless, there are two lines of research pertaining to the syntactic analysis of queries.

Linguistic analysis of queries. First, linguistic analysis and annotation of queries at the level of segmentation or chunking [43] and parts-of-speech tagging [78, 79, 80, 81] have been important directions of research since the early 2000s [8]. While these studies reveal interesting syntactic properties and trends, such as more than 70% of the query terms are nouns [78], and NL question queries are on the rise [82], they are based on the fundamental assumption that queries issued using the words of a certain parent language, say Standard English, will borrow grammatical artefacts of that parent language (i.e., nouns, verbs, noun and verb phrases, etc.). This assumption is biased because a noun in English is called a noun because it follows a particular syntactic distribution; it is quite unlikely that the same word will behave as a noun in a query either from the point of statistical distribution or its cognitive interpretation by the users. Thus, if queries are to be understood linguistically, they should be analyzed from the first principles rather than superimposing the grammatical syntax of NLs [83], which masks their *true* syntactic properties. We note that techniques based on the superimposition of syntax from the parent NL can still be useful for practical applications, but they cannot tell us much about the true syntax of Web search queries.

Entities and intents. The second line of research, which we believe is more promising, is the analysis of queries in terms of user intents. Such studies have looked into queries from various perspectives and have come up with various concepts such as entities and attributes [84, 85, 86, 87, 88], kernel-objects and modifiers [89] and query facets [90, 91], to factor the parts of a query and place it within a taxonomy of semantic or syntactic patterns. While it is not possible to review all these studies here, a closer look at the actual network motifs of the WCN for real queries reveals interesting synergy between the concepts of *intent words* [92, 93, 72] (also called modifiers or attributes) and *content words* (or entities or kernel-objects), which is worth mentioning here.

7. Conclusions

In this paper, we have tried to understand the syntactic complexity of Web search queries, a distinct mode of interaction between man and man-made systems. We have adopted a three-pronged approach: applying statistical language models (using *n*-grams and *n*-terms), using complex network modeling (with WCNs), and asking native speakers (Web search users). Our results underline the necessity of using multiple independent perspectives. Having entropy or perplexity similar to or lower than NL need not, by itself, be indicative of an underlying language system [27]. Network analysis shows bigrams to be within striking distance of replicating real log syntax at a corpus-level. However, when native speakers are consulted, individual queries generated by trigrams are found to be much more acceptable than those by bigrams. Only a combined approach is successful in bringing out the complete picture of *n*-gram-based statistics being inadequate, and the need for a language model that imbibes syntactic *and* semantic constraints specific to Web search queries. More interestingly, in both network and user experiments, a common behavior emerges: the results are distinct both from scenarios that assume queries being random word sequences *or* following the syntactic constraints of the parent natural language. We believe that these can indeed be considered positive cues in favor of acceptance of our original hypothesis of queries evolving into a distinct linguistic system. However, we do not claim that this research is complete by itself. Rather, it is only the first step towards a more holistic goal – when queries, communicating information needs of millions of users, can be established to be an independent language system from all the three aspects – structure, function and dynamics.

8. Acknowledgments

The first author was supported by Microsoft Corporation and Microsoft Research India under the Microsoft Research India PhD Fellowship Award (Grant No. 8901901/IITKGP-Fship-2011). We would like to thank Rohan Ramanath from R. V. College of Engineering (RVCE), Bengaluru, India, for helping us in posting tasks on AMT, Parantapa Bhattacharya from Indian Institute of Technology (IIT) Kharagpur, India, and Pavan Kumar from National Institute of Technology (NIT) Durgapur, India, for helping us with the experiments on network stability and motif analysis.

References

[1] B. J. Jansen, A. Spink, How are we searching the World Wide Web? A comparison of nine search engine transaction logs, Information Processing and Management 42 (1) (2006) 248 – 263. doi:http://dx.doi.org/10.1016/j.ipm.2004.10.007.
URL http://www.sciencedirect.com/science/article/pii/S0306457304001396

[2] C. Schwartz, Web search engines, Journal of the American Society for Information Science 49 (11) (1998) 973–982. doi:10.1002/(SICI)1097-4571(1998)49:11<973::AID-ASI3>3.0.CO;2-Z.
URL http://dx.doi.org/10.1002/(SICI)1097-4571(1998)49:11<973::AID-ASI3>3.0.CO;2-Z

[3] K. M. Risvik, R. Michelsen, Search engines and web dynamics, Computer Networks 39 (3) (2002) 289 – 302. doi:http://dx.doi.org/10.1016/S1389-1286(02)00213-X.
URL http://www.sciencedirect.com/science/article/pii/S138912860200213X

[4] A. Ntoulas, J. Cho, C. Olston, What's new on the web?: The evolution of the web from a search engine perspective, in: Proceedings of the 13th International Conference on World Wide Web, WWW '04, ACM, New York, NY, USA, 2004, pp. 1–12. doi:10.1145/988672.988674.
URL http://doi.acm.org/10.1145/988672.988674

[5] A. Spink, D. Wolfram, M. B. J. Jansen, T. Saracevic, Searching the web: the public and their queries, Journal of the American Society for Information Science and Technology 52 (2001) 226–234. doi:10.1002/1097-4571(2000)9999:9999<::AID-ASI1591>3.3.CO;2-I.
URL http://dl.acm.org/citation.cfm?id=362968.362979

[6] G. Pass, A. Chowdhury, C. Torgeson, A picture of search, in: Proceedings of the 1st international conference on Scalable information systems, InfoScale '06, ACM, New York, NY, USA, 2006, pp. 1–7. doi:10.1145/1146847.1146848.
URL http://doi.acm.org/10.1145/1146847.1146848

[7] R. Saha Roy, M. Choudhury, K. Bali, Are web search queries an evolving protolanguage?, in: Proceedings of the 9th International Conference on the Evolution of Language, Evolang 9, World Scientific Publishing Co., Singapore, 2012, pp. 304–311.

[8] B. J. Jansen, A. Spink, T. Saracevic, Real life, real users, and real needs: a study and analysis of user queries on the web, Inf. Process. Manage. 36 (2) (2000) 207–227. doi:10.1016/S0306-4573(99)00056-4.
URL http://dx.doi.org/10.1016/S0306-4573(99)00056-4

[9] E. Guichard, L'internet: Mesures des appropriations d'une technique intellectuelle, These, Ecole des hautes études en sciences sociales (Oct 2002).
URL http://tel.archives-ouvertes.fr/tel-00294711/en/

[10] J.-L. Dessalles, Du protolangage au langage : modèle d'une transition, Marges linguistiques 11 (2006) 142–152.

[11] R. Saha Roy, M. D. Reddy, N. Ganguly, M. Choudhury, Understanding the Linguistic Structure and Evolution of Web Search Queries, in: Proceedings of the 10th International Conference on the Evolution of Lan-

guage, Evolang X, World Scientific Publishing Co., Singapore, 2014, pp. 286–293.

[12] P. F. Brown, P. V. Desouza, R. L. Mercer, V. J. D. Pietra, J. C. Lai, Class-based n-gram models of natural language, Computational linguistics 18 (4) (1992) 467–479.

[13] R. Rao, N. Yadav, M. Vahia, H. Joglekar, R. Adhikari, I. Mahadevan, Entropic evidence for linguistic structure in the indus script, Science 324 (5931) (2009) 1165–1165.

[14] J. Gauvain, A. Messaoudi, H. Schwenk, Language recognition using phone lattices, in: Proceedings of the 8th International Conference on Spoken Language Processing, Vol. 4 of ICSLP '04, ISCA, France, 2004, pp. 1283–1286.

[15] J. Downie, Evaluating a simple approach to music information retrieval: Conceiving melodic n-grams as text, Ph.D. thesis, The University of Western Ontario (1999).

[16] R. Mantegna, S. Buldyrev, A. Goldberger, S. Havlin, C. Peng, M. Simons, H. Stanley, Systematic analysis of coding and noncoding dna sequences using methods of statistical linguistics, Physical Review E 52 (3) (1995) 2939.

[17] J. Bellegarda, Statistical language model adaptation: review and perspectives, Speech communication 42 (1) (2004) 93–108.

[18] M. Zissman, E. Singer, Automatic language identification of telephone speech messages using phoneme recognition and n-gram modeling, in: Acoustics, Speech, and Signal Processing, 1994. ICASSP-94., 1994 IEEE International Conference on, Vol. i, IEEE, New York City, 1994, pp. I/305–I/308 vol.1. doi:10.1109/ICASSP.1994.389377.

[19] P. Koehn, Statistical machine translation, Vol. 11, Cambridge University Press, Cambridge, UK, 2010.

[20] H. Duan, B.-J. P. Hsu, Online spelling correction for query completion, in: Proceedings of the 20th international conference on World wide web, WWW '11, ACM, New York, NY, USA, 2011, pp. 117–126. doi:10.1145/1963405.1963425.
URL http://doi.acm.org/10.1145/1963405.1963425

[21] J. M. Ponte, W. B. Croft, A language modeling approach to information retrieval, in: Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '98, ACM, New York, NY, USA, 1998, pp. 275–281. doi:10.1145/290941.291008.
URL http://doi.acm.org/10.1145/290941.291008

[22] M. Choudhury, A. Mukherjee, The structure and dynamics of linguistic networks, in: Dynamics on and of Complex Networks, Springer, New York, USA, 2009, pp. 145–166.

[23] R. Ferrer-i-Cancho, R. V. Solé, The small world of human language, Proceedings of the Royal Society of London B 268 (1482) (2001) 2261–2265.

[24] S. N. Dorogovtsev, J. F. F. Mendes, Language as an evolving word web, Proceedings of the Royal Society of London B 268 (1485) (2001) 2603–2606.

[25] N. Chomsky, Syntactic structures, de Gruyter Mouton, Berlin, 2002.

[26] T. Paikeday, The native speaker is dead!: An informal discussion of a linguistic myth with Noam Chomsky and other linguists, philosophers, psychologists, and lexicographers, Paikeday Publishing Inc., Toronto, 1985.

[27] R. Sproat, Last words: Ancient symbols, computational linguistics, and the reviewing practices of the general science journals, Computational Linguistics 36 (3).

[28] S. Sinha, A. Izhar, R. Pan, B. Wells, Network analysis of a corpus of undeciphered indus civilization inscriptions indicates syntactic organization, Computer Speech & Language 25 (3) (2011) 639–654.

[29] M. Srikanth, R. Srihari, Biterm language models for document retrieval, in: Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '02, ACM, New York, NY, USA, 2002, pp. 425–426.

[30] X. Yan, J. Guo, Y. Lan, X. Cheng, A biterm topic model for short texts, in: Proceedings of the 22Nd International Conference on World Wide Web, WWW '13, International World Wide Web Conferences Steering Committee, Republic and Canton of Geneva, Switzerland, 2013, pp. 1445–1456.
URL http://dl.acm.org/citation.cfm?id=2488388.2488514

[31] J. J. Hull, S. N. Srihari, Experiments in text recognition with binary n-gram and viterbi algorithms, IEEE Trans. Pattern Anal. Mach. Intell. 4 (5) (1982) 520–530. doi:10.1109/TPAMI.1982.4767297.
URL http://dx.doi.org/10.1109/TPAMI.1982.4767297

[32] T. Dunning, Statistical identification of language, Computing Research Laboratory, New Mexico State University, New Mexico, USA, 1994.

[33] C. D. Manning, H. Schütze, Foundations of statistical natural language processing, MIT press, Cambridge, MA, 1999.

[34] G. Salton, A. Wong, C. S. Yang, A vector space model for automatic indexing, Commun. ACM 18 (11) (1975) 613–620. doi:10.1145/361219.361220.
URL http://doi.acm.org/10.1145/361219.361220

[35] G. Salton, C. Buckley, Term-weighting approaches in automatic text retrieval, Inf. Process. Manage. 24 (5) (1988) 513–523. doi:10.1016/0306-4573(88)90021-0.
URL http://dx.doi.org/10.1016/0306-4573(88)90021-0

[36] F. Song, W. B. Croft, A general language model for information retrieval, in: Proceedings of the eighth international conference on Information and knowledge management, CIKM '99, ACM, New York, NY, USA, 1999, pp. 316–321. doi:10.1145/319950.320022.
URL http://doi.acm.org/10.1145/319950.320022

[37] D. Guthrie, B. Allison, W. Liu, L. Guthrie, Y. Wilks, A closer look at skip-gram modelling, in: Proceedings of the 5th international Conference on Language Resources and Evaluation, LREC '06, 2006, pp. 1–4.

[38] D. Metzler, W. B. Croft, A markov random field model for term dependencies, in: Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval, SIGIR '05, ACM, New York, NY, USA, 2005, pp. 472–479. doi:10.1145/1076034.1076115.
URL http://doi.acm.org/10.1145/1076034.1076115

[39] C. Biemann, S. Roos, K. Weihe, Quantifying semantics using complex network analysis, COLING '12 (December 2012).

[40] C. Biemann, L. Krumov, S. Roos, K. Weihe, Network motifs are a powerful tool for semantic distinction, in: A. Mehler, A. Lücking, S. Banisch, P. Blanchard, B. Job (Eds.), Towards a Theoretical Framework for Analyzing Complex Linguistic Networks, Understanding Complex Systems, Springer Berlin Heidelberg, 2016, pp. 83–105.
URL http://dx.doi.org/10.1007/978-3-662-47238-5_4

[41] B. Tan, F. Peng, Unsupervised query segmentation using generative language models and wikipedia, in: Proceedings of the 17th international conference on World Wide Web, WWW '08, ACM, New York, NY, USA, 2008, pp. 347–356. doi:10.1145/1367497.1367545.
URL http://doi.acm.org/10.1145/1367497.1367545

[42] Y. Li, B.-J. P. Hsu, C. Zhai, K. Wang, Unsupervised query segmentation using clickthrough for information retrieval, in: Proceedings of the 34th international ACM SIGIR conference on Research and development in Information Retrieval, SIGIR '11, ACM, New York, NY, USA, 2011, pp. 285–294. doi:10.1145/2009916.2009957.
URL http://doi.acm.org/10.1145/2009916.2009957

[43] M. Hagen, M. Potthast, A. Beyer, B. Stein, Towards optimum query segmentation: in doubt without, in: Proceedings of the 21st ACM international conference on Information and knowledge management, CIKM '12, ACM, New York, NY, USA, 2012, pp. 1015–1024. doi:10.1145/2396761.2398398.
URL http://doi.acm.org/10.1145/2396761.2398398

[44] R. Saha Roy, N. Ganguly, M. Choudhury, S. Laxman, An IR-based Evaluation Framework for Web Search Query Segmentation, in: SIGIR '12, ACM, 2012, pp. 881–890.

[45] L. Bahl, F. Jelinek, R. Mercer, A maximum likelihood approach to continuous speech recognition, Pattern Analysis and Machine Intelligence, IEEE Transactions on 5 (2) (1983) 179–190.

[46] P. Brown, V. Pietra, R. Mercer, S. Pietra, J. Lai, An estimate of an upper bound for the entropy of english, Computational Linguistics 18 (1) (1992) 31–40.

[47] R. Saha Roy, N. Ganguly, M. Choudhury, N. K. Singh, Complex network analysis reveals kernel-periphery structure in web search queries, in: QRU '11, ACM, New York, NY, USA, 2011, pp. 5–8.

[48] S. H. Strogatz, Exploring complex networks, Nature 410 (6825) (2001) 268–276.

[49] R. Albert, A.-L. Barabási, Statistical mechanics of complex networks, Reviews of modern physics 74 (1) (2002) 47.

[50] M. E. J. Newman, The structure and function of complex networks, SIAM Review 45 (2003) 167–256.

[51] A. Mehler, Large text networks as an object of corpus linguistic stud-

ies, in: Corpus Linguistics. An International Handbook of the Science of Language and Society, De Gruyter, Berlin, Germany, 2008, pp. 328–382.

[52] M. Choudhury, D. Chatterjee, A. Mukherjee, Global topology of word co-occurrence networks: Beyond the two-regime power-law, in: Proceedings of the 23rd International Conference on Computational Linguistics: Posters, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, pp. 162–170.

[53] R. Blanco, C. Lioma, Graph-based term weighting for information retrieval, Information Retrieval 15 (2012) 54–92. doi:$10.1007/s10791-011-9172-x$.
URL http://dx.doi.org/$10.1007/s10791-011-9172-x$

[54] D. J. Watts, S. H. Strogatz, Collective dynamics of 'small-world' networks, Nature 393 (6684) (1998) 440–442.

[55] P. Erdös, A. Rényi, On random graphs I, Publicationes Mathematicae Debrecen 5 (1959) 290–297.

[56] V. Batagelj, A. Mrvar, Pajek: Analysis and visualization of large networks, Graph Drawing 2265 (2002) 8–11.

[57] R. Milo, S. Shen-Orr, S. Itzkovitz, N. Kashtan, D. Chklovskii, U. Alon, Network motifs: Simple building blocks of complex networks, Science 298 (5594) (2002) 824–827. arXiv:http://www.sciencemag.org/content/298/5594/824.full.pdf.
URL http://www.sciencemag.org/content/298/5594/824.abstract

[58] N. Kashtan, S. Itzkovitz, R. Milo, U. Alon, Efficient sampling algorithm for estimating subgraph concentrations and detecting network motifs, Bioinformatics 20 (11) (2004) 1746–1758.

[59] F. Schreiber, H. Schwöbbermeyer, Mavisto: a tool for the exploration of network motifs, Bioinformatics 21 (17) (2005) 3572–3574.

[60] S. Wernicke, A faster algorithm for detecting network motifs, in: Proceedings of the 5th International conference on Algorithms in Bioinformatics, WABI'05, Springer-Verlag, Berlin, Heidelberg, 2005, pp. 165–177. doi:$10.1007/11557067_14$.
URL http://dx.doi.org/$10.1007/11557067_14$

[61] Z. R. M. Kashani, H. Ahrabian, E. Elahi, A. Nowzari-Dalini, E. S. Ansari, S. Asadi, S. Mohammadi, F. Schreiber, A. Masoudi-Nejad, Kavosh: a new algorithm for finding network motifs, BMC Bioinformatics 10 (2009) 318.

[62] S. Kullback, R. A. Leibler, On Information and Sufficiency, The Annals of Mathematical Statistics 22 (1) (1951) 79–86.

[63] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, New York, NY, USA, 2008.

[64] M. Heilman, N. A. Smith, Good question! statistical ranking for question generation, in: Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics, HLT '10, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, pp. 609–617.
URL http://dl.acm.org/citation.cfm?id=1857999.1858085

[65] R. Mitkov, L. An Ha, N. Karamanis, A computer-aided environment for generating multiple-choice test items, Nat. Lang. Eng. 12 (2) (2006) 177–194. doi:10.1017/S1351324906004177.
URL http://dx.doi.org/10.1017/S1351324906004177

[66] C. Shannon, Prediction and entropy of printed english, Bell System Technical Journal 30 (1) (1951) 50–64.

[67] B. Carterette, P. Bennett, D. Chickering, S. Dumais, Here or There: Here or There: Preference Judgments for Relevance, in: C. Macdonald, I. Ounis, V. Plachouras, I. Ruthven, R. White (Eds.), Advances in Information Retrieval, Vol. 4956 of Lecture Notes in Computer Science, Springer, Berlin, Heidelberg, 2008, pp. 16–27. doi:$10.1007/978-3-540-78646-7_5$.
URL http://dx.doi.org/$10.1007/978-3-540-78646-7_5$

[68] H. Li, G. Xu, W. B. Croft, M. Bendersky, Z. Wang, E. Viegas, QRU-1: A Public Dataset for Promoting Query Representation and Understanding Research, WSCD '12 (February 2012).

[69] O. Alonso, R. Baeza-Yates, Design and implementation of relevance assessments using crowdsourcing, in: Proceedings of the 33rd European conference on Advances in information retrieval, ECIR'11, Springer-Verlag, Berlin, Heidelberg, 2011, pp. 153–164.
URL http://dl.acm.org/citation.cfm?id=1996889.1996910

[70] V. R. Carvalho, M. Lease, E. Yilmaz, Crowdsourcing for search evaluation, SIGIR Forum 44 (2) (2011) 17–22. doi:10.1145/1924475.1924481.
URL http://doi.acm.org/10.1145/1924475.1924481

[71] M. Hagen, M. Potthast, B. Stein, C. Bräutigam, Query segmentation revisited, in: Proceedings of the 20th international conference on World wide web, WWW '11, ACM, New York, NY, USA, 2011, pp. 97–106. doi:10.1145/1963405.1963423.
URL http://doi.acm.org/10.1145/1963405.1963423

[72] R. Saha Roy, R. Katare, N. Ganguly, S. Laxman, M. Choudhury, Discovering and understanding word level user intent in Web search queries, Web Semantics: Science, Services and Agents on the World Wide Web 30 (2015) 22–38.

[73] B. Zhao, M. Eck, S. Vogel, Language model adaptation for statistical machine translation with structured query models, in: Proceedings of the 20th international conference on Computational Linguistics, COLING '04, Association for Computational Linguistics, Stroudsburg, PA, USA, 2004, pp. 1–7. doi:10.3115/1220355.1220414.
URL http://dx.doi.org/10.3115/1220355.1220414

[74] J. Philbin, O. Chum, M. Isard, J. Sivic, A. Zisserman, Object retrieval with large vocabularies and fast spatial matching, in: Computer Vision and Pattern Recognition, 2007. CVPR'07. IEEE Conference on, IEEE, Los Alamitos, CA, USA, 2007, pp. 1–8.

[75] W. Croft, D. Metzler, T. Strohman, Search engines: Information retrieval in practice, Addison-Wesley, Boston, USA, 2010.

[76] M. D. Hauser, N. Chomsky, W. T. Fitch, The faculty of language: What is it, who has it, and how did it evolve?, science 298 (5598) (2002) 1569–1579.

[77] D. Biber, E. Finegan, Sociolinguistic perspectives on register, Oxford University Press, 1994.

[78] C. Barr, R. Jones, M. Regelson, The linguistic structure of english web-search queries, in: Proceedings of the Conference on Empirical Methods in Natural Language Processing, EMNLP '08, Association for Computational Linguistics, Stroudsburg, PA, USA, 2008, pp. 1021–1030.
URL http://dl.acm.org/citation.cfm?id=1613715.1613848

[79] M. Bendersky, W. B. Croft, D. A. Smith, Structural annotation of search queries using pseudo-relevance feedback, in: Proceedings of the 19th ACM international conference on Information and knowledge management, CIKM '10, ACM, New York, NY, USA, 2010, pp. 1537–1540. doi:10.1145/1871437.1871666.
URL http://doi.acm.org/10.1145/1871437.1871666

[80] M. Bendersky, W. B. Croft, D. A. Smith, Joint annotation of search queries, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1, HLT '11, Association for Computational Linguistics, Stroudsburg, PA, USA, 2011, pp. 102–111.
URL http://dl.acm.org/citation.cfm?id=2002472.2002486

[81] K. Ganchev, K. Hall, R. McDonald, S. Petrov, Using search-logs to improve query tagging, in: Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2, ACL '12, Association for Computational Linguistics, Stroudsburg, PA, USA, 2012, pp. 238–242.
URL http://dl.acm.org/citation.cfm?id=2390665.2390723

[82] B. Pang, R. Kumar, Search in the lost sense of "query": question formulation in web search queries and its temporal changes, in: Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2, HLT '11, Association for Computational Linguistics, Stroudsburg, PA, USA, 2011, pp. 135–140.
URL http://dl.acm.org/citation.cfm?id=2002736.2002766

[83] N. Mishra, R. Saha Roy, N. Ganguly, S. Laxman, M. Choudhury, Unsupervised query segmentation using only query logs, in: Proceedings of the 20th international conference companion on World wide web, WWW '11, ACM, New York, NY, USA, 2011, pp. 91–92. doi:10.1145/1963192.1963239.
URL http://doi.acm.org/10.1145/1963192.1963239

[84] M. Pasca, Acquisition of categorized named entities for web search, in: Proceedings of the thirteenth ACM international conference on Information and knowledge management, ACM, New York, NY, USA, 2004, pp. 137–145.

[85] M. Paşca, B. Van Durme, What you seek is what you get: extraction of class attributes from query logs, in: Proceedings of the 20th international joint conference on Artifical intelligence, IJCAI'07, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2007, pp. 2832–2837.

URL http://dl.acm.org/citation.cfm?id=1625275.1625731

[86] J. Reisinger, M. Paşca, Low-cost supervision for multiple-source attribute extraction, in: Proceedings of the 10th International Conference on Computational Linguistics and Intelligent Text Processing, CICLing '09, Springer-Verlag, Berlin, Heidelberg, 2009, pp. 382–393. doi:http://dx.doi.org/$10.1007/978-3-642-00382-0_31$.
URL http://dx.doi.org/$10.1007/978-3-642-00382-0_31$

[87] E. Alfonseca, M. Paşca, E. Robledo-Arnuncio, Acquisition of instance attributes via labeled and related instances, in: SIGIR '10, 2010, pp. 58–65.

[88] A. Jain, M. Pennacchiotti, Open entity extraction from web search query logs, in: Proceedings of the 23rd International Conference on Computational Linguistics, COLING '10, Association for Computational Linguistics, Stroudsburg, PA, USA, 2010, pp. 510–518.
URL http://dl.acm.org/citation.cfm?id=1873781.1873839

[89] H. Yu, F. Ren, Role-explicit query identification and intent role annotation, in: Proceedings of the 21st ACM international conference on Information and knowledge management, CIKM '12, ACM, New York, NY, USA, 2012, pp. 1163–1172. doi:10.1145/2396761.2398416.
URL http://doi.acm.org/10.1145/2396761.2398416

[90] C. González-Caro, R. Baeza-Yates, A multi-faceted approach to query intent classification, in: R. Grossi, F. Sebastiani, F. Silvestri (Eds.), SPIRE '11, Vol. 7024 of Lecture Notes in Computer Science, Springer, Berlin, 2011, pp. 368–379.

[91] V. B. Nguyen, M. Y. Kan, Functional faceted web query analysis, in: Proceedings of the Workshop on Query Log Analysis: Social And Technological Challenges, WWW '07, ACM, New York, NY, USA, 2007, pp. 1–8.

[92] T. Lin, P. Pantel, M. Gamon, A. Kannan, A. Fuxman, Active objects: Actions for entity-centric search, in: Proceedings of the 21st international conference on World Wide Web, WWW '12, ACM, New York, NY, USA, 2012, pp. 589–598. doi:10.1145/2187836.2187916.
URL http://doi.acm.org/10.1145/2187836.2187916

[93] X. Yin, S. Shah, Building taxonomy of web search intents for name entity queries, in: Proceedings of the 19th international conference on World wide web, WWW '10, ACM, New York, NY, USA, 2010, pp. 1001–1010. doi:10.1145/1772690.1772792.
URL http://doi.acm.org/10.1145/1772690.1772792