Using Web N-Grams to Help Second-Language Speakers

Martin Potthast Martin Trenkmann Benno Stein

Bauhaus-Universität Weimar
99421 Weimar, Germany
<first name>.<last name>@uni-weimar.de

1. INTRODUCTION

We demonstrate the NETSPEAK search engine which helps people with writing in a foreign language.[1] To this end the search engine implements wildcard query processing on top of an inverted index of Web n-grams. By indexing word n-grams extracted from the whole Web, our search engine allows writers to retrieve phrases matching almost any given context, while the occurrence frequencies of the retrieved n-grams indicate their *commonness* in everyday writing. To allow for retrieval in less than a second for most queries, we have devised an inverted index and a query processor tailored to n-gram retrieval.

1.1 Related Work

Today's word processors offer tools to check the correctness of a phrase, but they provide no evidence about text commonness, so that writers are left in doubt whether something just written is in fact how others would write it. Therefore they turn to Web search engines for such kinds of checks, using (quoted) phrase queries. This strategy, however, is questionable since the occurrence frequencies reported by search engines are inexact and often overestimate the true figures by far. If a search engine allows for wildcards, the top-ranked result snippets often contain only the most frequently used phrase that matches the query, and the user is left with piecing together other matching alternatives. In computer linguistics wildcard search engines similar to ours have been set up for research purposes [3, 4, 5]. Though the use cases of these search engines differ from ours, the underlying problems are quite similar: retrieval speed and scalability. Informally, our search engine outperforms the available prototypes, but a rigorous comparison is rendered difficult without re-implementing existing approaches. Finally, Web n-grams are being used to construct language models. Our tool can be used for this purpose; however, there are approaches more specialized to language model indexing, e.g. [2].

2. WILDCARD N-GRAM RETRIEVAL

The heart of NETSPEAK is an index of n-grams—currently the Google n-grams [1]. The index maps the vocabulary of words found in the n-gram collection onto lists of n-gram references. Each n-gram reference consists of three values: a pointer to the byte position of an n-gram on the hard disk, the word position of the indexed word within the n-gram, and the frequency count of the n-gram.

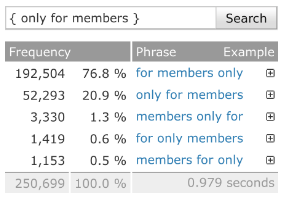
Our wildcard query language currently provides a number of basic operators that match any number of words (asterisk sign, $*$), exactly one word (question mark, ?), any order of words (words enclosed by curly brackets, `{.}`), any one of a set of words (word enclosed by square brackets, `[.]`), and any of a word's synonyms (the tilde sign in front of a word, ~). Valid queries are composed of at least one literal word and any number of operators. A query is processed by retrieving and intersecting the postlists of the literal words in the query, and by filtering those n-grams that do not match the query pattern. The query processor does not retrieve the actual n-gram strings but determines matching n-grams solely on the basis of n-gram references, which speeds up retrieval by minimizing random disk accesses. Note that all of the aforementioned wildcard search engines resort to post-retrieval string matching instead. Moreover, our query processor retrieves only the top-k matching n-grams using a search pruning heuristic; details can be found in [6].

[1]NETSPEAK is freely accessible at www.netspeak.cc

SIGIR 2010 WEB N-GRAM Workshop, July 23, 2010, Geneva, Switzerland.

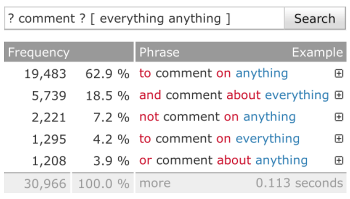
3. USER INTERFACE

The figures below show the result sets of two queries: left it is searched for any order of the three words "for members only", right it is searched for 4-grams whose second word is "comment" and whose fourth word is either "everything" or "anything". The frequency column report for each n-gram the well-known values support and confidence. The former gives an idea about absolute usage while the latter informs about relative usage compared to all n-grams that match the query. In practice, neither of the values can go without the other. Also note that the most frequent n-gram retrieved is not necessarily the "right" n-gram to write. A writer still has to determine which n-grams actually fit her context and to decide which one to use, but with our search engine, this decision becomes for the first time an informed decision. If unsure about how to include an n-gram into a sentence, a click on the plus sign next to each n-gram shows a couple of examples.



{ only for members } Search

Frequency		Phrase	Example
192,504	76.8 %	for members only	⊞
52,293	20.9 %	only for members	⊞
3,330	1.3 %	members only for	⊞
1,419	0.6 %	for only members	⊞
1,153	0.5 %	members for only	⊞
250,699	100.0 %		0.979 seconds



? comment ? [everything anything] Search

Frequency		Phrase	Example
19,483	62.9 %	to comment on anything	⊞
5,739	18.5 %	and comment about everything	⊞
2,221	7.2 %	not comment on anything	⊞
1,295	4.2 %	to comment on everything	⊞
1,208	3.9 %	or comment about anything	⊞
30,966	100.0 %	more	0.113 seconds

4. REFERENCES

[1] T. Brants and A. Franz. Web 1T 5-gram Version 1. Linguistic Data Consortium LDC2006T13, Philadelphia, 2006.

[2] T. Brants, A. C. Popat, P. Xu, F. J. Och, and J. Dean. Large Language Models in Machine Translation. In *EMNLP-CoNLL'07*.

[3] M. J. Cafarella and O. Etzioni. A Search Engine for Natural Language Applications. In *WWW'05*.

[4] P. Resnik and A. Elkiss. The Linguist's Search Engine: An Overview. In *ACL'05*.

[5] S. Sekine. A Linguistic Knowledge Discovery Tool: Very Large Ngram Database Search with Arbitrary Wildcards. In *COLING'08*.

[6] B. Stein, M. Potthast, and M. Trenkmann. Retrieving Customary Web Language to Assist Writers. In *ECIR'10*.