Google Web 1T 5-Grams Made Easy (but not for the computer)

Stefan Evert
Institute of Cognitive Science
University of Osnabrück
49069 Osnabrück, Germany
stefan.evert@uos.de

Abstract

This paper introduces *Web1T5-Easy*, a simple indexing solution that allows interactive searches of the Web 1T 5-gram database and a derived database of quasi-collocations. The latter is validated against co-occurrence data from the BNC and ukWaC on the automatic identification of non-compositional VPC.

1 Introduction

The *Google Web 1T 5-gram* (Web1T5) database (Brants and Franz, 2006) consists of frequency counts for bigram, trigrams, 4-grams and 5-grams extracted from 1 trillion words of English Web text, i.e. from a corpus 10,000 times the size of the British National Corpus (Aston and Burnard, 1998). While primarily designed as a resource to build better language models for machine translation and other NLP applications, its public release in 2006 was greeted with great enthusiasm by many researchers in computational linguistics. As one example, Mitchell et al. (2008) used the Web1T5 data successfully to predict fMRI neural activation associated with concrete noun concepts.

For linguistic applications, though, the Web1T5 database presents three major obstacles:

(i) The lack of linguistic annotation: Google's tokenisation splits hyphenated compounds (e.g., *part-time* is split into a three-token sequence *part*|-|*time*) and differs in many other ways from the rules used in liguistic corpora. The n-grams are neither annotated with part-of-speech tags nor lemmatised, and there are separate entries for sentence-initial uppercase and the corresponding lowercase forms.

(ii) The application of frequency thresholds: Despite the enormous size of the database, its compilers found it necessary to omit low-frequency n-grams with fewer than 40 occurrences. This means that non-adjacent word combinations are listed only if the occur in a relatively frequent pattern. As a consequence, it is impossible to obtain reliable frequency estimates for latent phenomena by pooling data (e.g. the co-occurrence frequency of a particular verb with nouns denoting animals).

(iii) The difficulty of interactive search: The complete Web1T5 database consists of 24.4 GiB of binary-sorted, compressed text files. While this format is suitable for building n-gram language models and other offline processing, searching the database is not efficient enough for interactive use. Except for simple, case-sensitive prefix searches – which can be restricted to a single file containing 50–90 MiB of compressed text – every query requires a linear scan of the full database.

This paper presents a simple open-source software solution to the third problem, called *Web1T5-Easy*. The n-gram data are encoded and indexed in a relational database. Building on convenient open-source tools such as SQLite and Perl, the software aims to strike a good balance between search efficiency and ease of use and implementation. With its focus on interactive, but accurate search it complements the approximate indexing and batch processing approaches of Hawker et al. (2007). *Web1T5-Easy* can be downloaded from `http://webascorpus.sf.net/Web1T5-Easy/`.[1]

[1] An online demo of the complete Web1T5 database is available at `http://cogsci.uos.de/~korpora/ws/Web1T5/`.

word 1	*word 2*	*word 3*	*f*
supplement	depend	on	193
supplement	depending	on	174
supplement	depends	entirely	94
supplement	depends	on	338
supplement	derived	from	2668
supplement	des	coups	77
supplement	described	in	200

Table 1: Example of Web1T5 3-gram frequency data (excerpt from file `3gm-0088.gz`).

The rest of this paper is organised as follows. Section 2 describes the general system architecture in more detail. Section 3 explains how collocations (with a maximal span size of four tokens) and distributional semantic models (DSM) can be approximated on the basis of Web1T5 frequency data. Some technical aspects are summarised in Section 4. Section 5 addresses the consequences of problems (i) and (ii). The linguistic usefulness of Web1T5 collocation data is validated on a multiword extraction task from the MWE 2008 workshop.[2] Section 6 concludes with a brief outlook on the future development of Web1T5-Easy.

2 System architecture

While designing the fastest possible indexing architecture for the Web1T5 database is an interesting computer science problem, linguistic applications typically do not require the millisecond response times of a commercial search engine. It is sufficient for interactive queries to be completed within a few seconds, and many users will also be willing to wait several minutes for the result of a complex search operation. Given the tabular format of the Web1T5 n-gram frequency data (cf. Table 1), it was a natural choice to make use of a standard relational database (RDBMS). Database tables can be indexed on single or multiple columns for fast access, and the SQL query languge allows flexible analysis and aggregation of frequency data (see Section 2.2 for some examples). While the indexing procedure can be very time-consuming, it is carried out offline and has to run only once.

[2] `http://multiword.sf.net/mwe2008/`

Web1T5-Easy was designed to balance computational efficiency against implementation effort and ease of use. Its main ingredients are the public-domain embedded relational database engine SQLite and the open-source scripting language Perl which are connected through the portable DBI/DBD interface.[3] The Web1T5-Easy package consists of two sets of Perl scripts. The first set automates pre-processing and indexing, detailed in Section 2.1. The second set, which facilitates command-line access to the database and provides a Web-based GUI, is described in Section 2.2. Technical details of the representation format and performance figures are presented in Section 4.

The embedded database engine SQLite was preferred over a full-fledged RDBMS such as MySQL or PostgreSQL for several reasons: (i) running the database as a user-level process gives better control over huge database files and expensive indexing operations, which might otherwise clog up a dedicated MySQL server computer; (ii) each SQLite database is stored in a single, platform-independent file, so it can easily be copied to other locations or servers; (iii) an embedded database avoids the overhead of exchanging large amounts of data between client and server; (iv) tight integration with the application program allows more flexible use of the database than pure SQL queries (e.g., a Perl script can define its own SQL functions, cf. Section 3).

It is quite possible that the sophisticated query optimisers of MySQL and commercial RDMBS implementations would improve performance on complex SQL queries. Since Web1T5-Easy uses the generic DBI interface, it can easily be adapted to any RDMBS back-end for which DBI/DBD drivers are available.

2.1 The indexing procedure

Indexing of the Web1T5 n-gram data is carried out in four stages:

1. In an optional pre-processing step, words are filtered and normalised to lowercase.[4] Each

[3] See the Web pages at `http://www.sqlite.org/`, `http://www.perl.org/` and `http://dbi.perl.org/`.

[4] The default filter replaces numbers by the code `NUM`, various punctuation symbols by the code `PUN`, and all "messy" strings by the code `UNK`. It can easily be replaced by a user-defined normalisation mapping.

word in an n-gram entry is then coded as a numeric ID, which reduces database size and improves both indexing and query performance (see Section 4 for details on the representation format). The resulting tuples of $n+1$ integers (n word IDs plus frequency count) are inserted into a database table.

2. If normalisation was applied, the table will contain multiple entries for many n-grams.[5] In Stage 2, their frequency counts are aggregated with a suitable SQL query. This is one of the most expensive and disk-intensive operations of the entire indexing procedure.

3. A separate SQL index is created for each n-gram position (e.g., *word 1*, *word 2* and *word 3* in Table 1). Multi-column indexes are currently omitted, as they would drastically increase the size of the database files.[6] Moreover, the use of an index only improves query execution speed if it is highly selective, as explained in Section 4. If desired, the Perl scripts can trivially be extended to create additional indexes.

4. A statistical analysis of the database is performed to improve query optimisation (i.e., appropriate selection of indexes).

The indexing procedure is carried out separately for bigrams, trigrams, 4-grams and 5-grams, using a shared lexicon table to look up numeric IDs. Users who do not need the larger n-grams can easily skip them, resulting in a considerably smaller database and much faster indexing.

2.2 Database queries and the Web GUI

After the SQLite database has been populated and indexed, it can be searched with standard SQL queries (typically a join between one of the n-gram tables and the lexicon table), e.g. using the `sqlite3` command-line utility. Since this requires detailed knowledge of SQL syntax as well as the database layout and normalisation rules, the Web1T5-Easy package offers a simpler, user-friendly query language, which is internally translated into appropriate SQL code.

A Web1T5-Easy query consists of 2–5 search terms separated by blanks. Each search term is either a literal word (e.g. `sit`), a set of words in square brackets (e.g. `[sit,sits,sat,sitting]`), a prefix (`under%`) or suffix (`%ation`) expression, `*` for an arbitrary word, or `?` to skip a word. The difference between the latter two is that positions marked by `*` are included in the query result, while those marked by `?` are not. If a query term cannot match because of normalisation, an informative error message is shown. Matches can be ranked by frequency or by association scores, according to one of the measures recommended by Evert (2008): t-score (t), log-likelihood (G^2), chi-squared with Yates' correction (X^2), pointwise MI, or a version of the Dice coefficient.

For example, the query `web as corpus` shows that the trigram *Web as Corpus* occurs 1,104 times in the Google corpus (case-insensitive). `%ly good fun` lists ways of having fun such as *really good fun* (12,223×), *jolly good fun* (3,730×) and *extremely good fun* (2,788×). The query `[sit,sits,sat,sitting] * ? chair` returns the patterns *SIT in ... chair* (201,084×), *SIT on ... chair* (61,901×), *SIT at ... chair* (1,173×), etc. Corpus frequencies are automatically summed over all fillers in the third slot.

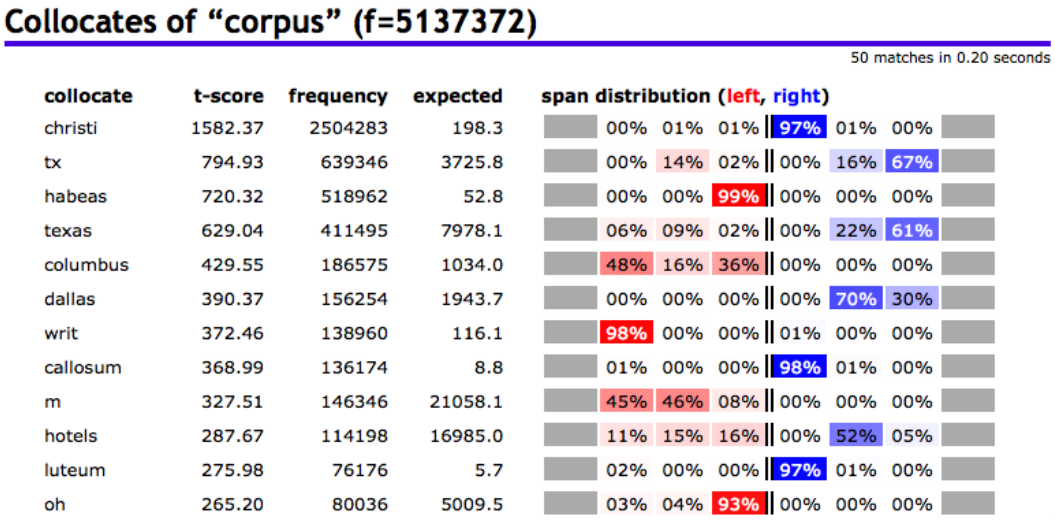
The query implementation is available as a command-line version and as a CGI script that provides a Web-based GUI to the Web1T5-Easy database. The CGI version also offers CSV and XML output formats for use as a Web service.

3 Quasi-collocations and DSM

Many corpus linguists and lexicographers will particularly be interested in using the Web1T5 database as a source of collocations (in the sense of Sinclair (1991)). While the British National Corpus at best provides sufficient data for a collocational analysis of some 50,000 words (taking $f \geq 50$ to be the minimum corpus frequency necessary), Web1T5 offers comprehensive collocation data for almost 500,000

[5] For example, with the default normalisation, *bought 2 bottles*, *bought 5 bottles*, *Bought 3 bottles*, *BOUGHT 2 BOTTLES* and many other trigrams are mapped to the representation *bought NUM bottles*. The database table thus contains multiple entries of the trigram *bought NUM bottles*, whose frequency counts have to be added up.

[6] For the 5-gram table, 10 different two-column indexes would be required to cover a wide range of queries, more than doubling the size of the database file.



Collocates of "corpus" (f=5137372)

50 matches in 0.20 seconds

collocate	t-score	frequency	expected	span distribution (left, right)
christi	1582.37	2504283	198.3	00% 01% 01% || 97% 01% 00%
tx	794.93	639346	3725.8	00% 14% 02% || 00% 16% 67%
habeas	720.32	518962	52.8	00% 00% 99% || 00% 00% 00%
texas	629.04	411495	7978.1	06% 09% 02% || 00% 22% 61%
columbus	429.55	186575	1034.0	48% 16% 36% || 00% 00% 00%
dallas	390.37	156254	1943.7	00% 00% 00% || 00% 70% 30%
writ	372.46	138960	116.1	98% 00% 00% || 01% 00% 00%
callosum	368.99	136174	8.8	01% 00% 00% || 98% 01% 00%
m	327.51	146346	21058.1	45% 46% 08% || 00% 00% 00%
hotels	287.67	114198	16985.0	11% 15% 16% || 00% 52% 05%
luteum	275.98	76176	5.7	02% 00% 00% || 97% 01% 00%
oh	265.20	80036	5009.5	03% 04% 93% || 00% 00% 00%

Figure 1: Quasi-collocations for the node word *corpus* in the Web GUI of Web1T5-Easy.

words (which have at least 50 different collocates in the database, and $f \geq 10{,}000$ in the original Google corpus).

Unfortunately, the Web1T5 distribution does not include co-occurrence frequencies of word pairs, except for data on immediately adjacent bigrams. It is possible, though, to derive approximate co-occurrence frequencies within a collocational span of up to 4 tokens. In this approach, each n-gram table yields information about a specific collocate position relative to the node. For instance, one can use the 4-gram table to identify collocates of the node word *corpus* at position $+3$ (i.e., 3 tokens to the right of the node) with the Web1T5-Easy query `corpus ? ? *`, and collocates at position -3 (i.e., 3 tokens to the left of the node) with the query `* ? ? corpus`. Co-occurrence frequencies within a collocational span, e.g. $(-3,+3)$, are obtained by summation over all collocate positions in this window, collecting data from multiple n-gram tables.

It has to be kept in mind that such *quasi-collocations* do not represent the true co-occurrence frequencies, since an instance of co-occurrence of two words is counted only if it forms part of an n-gram with $f \geq 40$ that has been included in Web1T5. Especially for larger distances of 3 or 4 tokens, this limitation is likely to discard most of the evidence for co-occurrence and put a focus on collocations that form part of a rigid multiword unit or institutionalised phrase. Thus, *cars* becomes the most salient collocate of *collectibles* merely because the two words appear in the slogan *from collectibles to cars* (9,443,572×). Section 5 validates the linguistic usefulness of Web1T5 quasi-collocations in a multiword extraction task.

Web1T5-Easy compiles frequency data for quasi-collocations in an additional step after the complete n-gram data have been indexed. For each pair of co-occurring words, the number of co-occurrences in each collocational position $(-4,-3,\ldots,+3,+4)$ is recorded. If the user has chosen to skip the largest n-gram tables, only a shorter collocational span will be available.

The Web GUI generates SQL code to determine co-occurrence frequencies within a user-defined collocational span on the fly, by summation over the appropriate columns of the quasi-collocations table. Collocates can be ranked by a range of association measures (t, G^2, X^2, MI, Dice, or frequency f), which are implemented as user-defined SQL functions in the Perl code. In this way, sophisticated statistical analyses can be performed even if they are not directly supported by the RDBMS back-end. Figure 1 shows an example of quasi-collocations in the Web GUI, ranked according to the t-score measure. On the right-hand side of the table, the distribution across collocate positions is visualised.

In computational linguistics, collocations play an important role as the term-term co-occurrence matrix underlying distributional semantic models

size (GiB)	*database file*	*no. of rows*
0.23	`vocabulary`	5,787,556
7.24	`2-grams`	153,634,491
32.81	`3-grams`	594,453,302
64.32	`4-grams`	933,385,623
75.09	`5-grams`	909,734,581
31.73	`collocations`	494,138,116
211.42	*total*	3,091,133,669

Table 2: Size of the fully indexed Web1T5 database, including quasi-collocations.

(DSM), with association scores used as feature weights (see e.g. Curran (2004, Sec. 4.3)). The Web1T5-Easy quasi-collocations table provides a sparse representation of such a term-term matrix, where only 494×10^6 or 0.0015% of the $5.8 \times 10^6 \cdot 5.8 \times 10^6 = 33.5 \times 10^{12}$ cells of a full co-occurrence matrix are populated with nonzero entries.

4 Technical aspects

An essential feature of Web1T5-Easy is the numeric coding of words in the n-gram tables, which allows for compact storage and more efficient indexing of the data than a full character string representation. A separate lexicon table lists every (normalised) word form together with its corpus frequency and an integer ID. The lexicon is sorted by decreasing frequency: since SQLite encodes integers in a variable-length format, it is advantageous to assign low ID numbers to the most frequent terms.

Every table is stored in its own SQLite database file, e.g. `vocabulary` for the lexicon table and `collocations` for quasi-collocations (cf. Section 3). The database files for different n-gram sizes (`2-grams`, `3-grams`, `4-grams`, `5-grams`) share the same layout and differ only in the number of columns. Table 2 lists the disk size and number of rows of each database file, with default normalisation applied. While the total size of 211 GiB by far exceeds the original Web1T5 distribution, it can easily be handled by modern commodity hardware and is efficient enough for interactive queries.

Performance measurements were made on a midrange 64-bit Linux server with 2.6 GHz AMD Opteron CPUs (4 cores) and 16 GiB RAM. SQLite database files and temporary data were stored on a fast, locally mounted hard disk. Similar or better hardware will be available at most academic institutions, and even in recent personal desktop PCs.

Indexing the n-gram tables in SQLite took about two weeks. Since the server was also used for multiple other memory- and disk-intensive tasks during this period, the timings reported here should only be understood as rough indications. The indexing process might be considerably faster on a dedicated server. Roughly equal amounts of time were spent on each of the four stages listed in Section 2.1.

Database analysis in Stage 4 turned out to be of limited value because the SQLite query optimiser was not able to make good use of this information. Therefore, a heuristic optimiser based on individual term frequencies was added to the Perl query scripts. This optimiser chooses the n-gram slot that is most likely to speed up the query, and explicitly disables the use of indexes for all other slots. Unless another constraint is much more selective, preference is always given to the first slot, which represents a clustered index (i.e. database rows are stored in index order) and can be scanned very efficiently.

With these explicit optimisations, Stage 4 of the indexing process can be omitted. If normalisation is not required, Stage 2 can also be skipped, reducing the total indexing time by half.

At first sight, it seems to be easy to compile the database of quasi-collocations one node at a time, based on the fully indexed n-gram tables. However, the overhead of random disk access during index lookups made this approach intractable.[7] A brute-force Perl script that performs multiple linear scans of the complete n-gram tables, holding as much data in RAM as possible, completed the compilation of co-occurrence frequencies in about three days.

Table 3 shows execution times for a selection of Web1T5-Easy queries entered in the Web GUI. In general, prefix queries that start with a reasonably specific term (such as `time of *`) are very fast, even on a cold cache. The query `%ly good fun` is a pathological case: none of the terms is selective enough to make good use of the corresponding

[7] In particular, queries like `* ? ? corpus` that scan for collocates to the left of the node word are extremely inefficient, since the index on the last n-gram slot is not clustered and accesses matching database rows in random order.

Web1T5-Easy query	*cold cache*	*warm cache*
`corpus linguistics`	0.11s	0.01s
`web as corpus`	1.29s	0.44s
`time of *`	2.71s	1.09s
`%ly good fun`	181.03s	24.37s
`[sit,sits,sat,sitting] * ? chair`	1.16s	0.31s
`* linguistics` *(association ranking)*	11.42s	0.05s
`university of *` *(association ranking)*	1.48s	0.48s
collocations of `linguistics`	0.21s	0.13s
collocations of `web`	6.19s	3.89s

Table 3: Performance of interactive queries in the Web GUI of Web1T5-Easy. Separate timings are given for a cold disk cache (first query) and warm disk cache (repeated query). Re-running a query with modified display or ranking settings will only take the time listed in the last column.

index, and entries matching the wildcard expression `%ly` in the first slot are scattered across the entire trigram table.

5 Validation on a MWE extraction task

In order to validate the linguistic usefulness of Web1T5 quasi-collocations, they were evaluated on the English VPC shared task from the MWE 2008 workshop.[8] This data set consists of 3,078 verb-particle constructions (VPC), which have been manually annotated as compositional or non-compositional (Baldwin, 2008). The task is to identify non-compositional VPC as true positives (TP) and re-rank the data set accordingly. Evaluation is carried out in terms of precision-recall graphs, using average precision (AP, corresponding to the area under a precision-recall curve) as a global measure of accuracy.

Frequency data from the Web1T5 quasi-collocations table was used to calculate association scores and rankings. Since previous studies suggest that no single association measure works equally well for all tasks and data sets, several popular measures were included in the evaluation: t-score (t), chi-squared with Yates' continuity correction (X^2), Dice coefficient (Dice), co-occurrence frequency (f), log-likelihood (G^2) and Mutual Information (MI); see e.g. Evert (2008) for full equations and references. The results are compared against rankings obtained from more traditional, linguistically annotated corpora of British English: the balanced, 100-million-word British National Corpus (Aston and Burnard, 1998) and the 2-billion-word Web corpus ukWaC (Baroni et al., 2009).

For BNC and ukWaC, three different extraction methods were used: (i) adjacent *bigrams* of verb + particle/preposition; (ii) shallow *syntactic patterns* based on POS tags (allowing pronouns and simple noun phrases between verb and particle); and (iii) surface co-occurrence within a *collocational span* of 3 tokens to the right of the node $(+1,+3)$, filtered by POS tag. Association scores were calculated using the same measures as for the Web1T5 quasi-collocations. Preliminary experiments with different collocational spans showed consistently lower accuracy than for $(+1,+3)$. In each case, the same association measures were applied as for Web1T5.

Evaluation results are shown in Figure 3 (graphs) and Table 4 (AP). The latter also describes the coverage of the corpus data by listing the number of candidates for which no frequency information is available (second column). These candidates are always ranked at the end of the list. While the BNC has a coverage of 92%–94% (depending on extraction method), scaling up to Web1T5 completely eliminates the missing data problem.

However, identification of non-compositional VPC with the Web1T5 quasi-collocations is considerably less accurate than with linguistically annotated data from the much smaller BNC. For recall values above 50%, the precision of statistical association measures such as t and X^2 is particularly poor

[8] `http://multiword.sf.net/mwe2008/`

	coverage	*average precision (%)*					
	(missing)	*t*	X^2	*Dice*	*f*	G^2	*MI*
BNC (bigrams)	242	**30.04**	29.75	27.12	26.55	29.86	22.79
BNC (syntactic patterns)	201	30.42	30.49	27.48	25.87	**30.64**	22.48
BNC (span $+1\ldots+3$)	185	29.15	**32.12**	30.13	24.33	31.06	22.58
ukWaC (bigrams)	171	29.28	**30.32**	27.79	25.37	29.63	25.13
ukWaC (syntactic patterns)	162	29.20	**31.19**	27.90	24.19	30.06	25.08
ukWaC (span $+1\ldots+3$)	157	27.82	**32.66**	30.54	23.03	30.01	25.76
Web1T5 (span $+1\ldots+3$)	3	**25.83**	25.27	25.33	20.88	25.77	20.81
BNC untagged ($+1\ldots+3$)	39	27.22	27.85	**28.98**	22.51	28.13	19.60

Table 4: Evaluation results for English non-compositional VPC (Baldwin, 2008): average precision (AP) as a global indicator. The baseline AP for random candidate ranking is 14.29%. The best result in each row is highlighted in bold.

(Figure 3.h). On the annotated corpora, where nodes and collocates are filtered by POS tags, best results are obtained with the least constrained extraction method and the chi-squared (X^2) measure. Scaling up to the 2-billion-word ukWaC corpus gives slightly better coverage and precision than on the BNC. Moreover, X^2 is now almost uniformly better than (or equal to) any other measure (Figure 3.f).

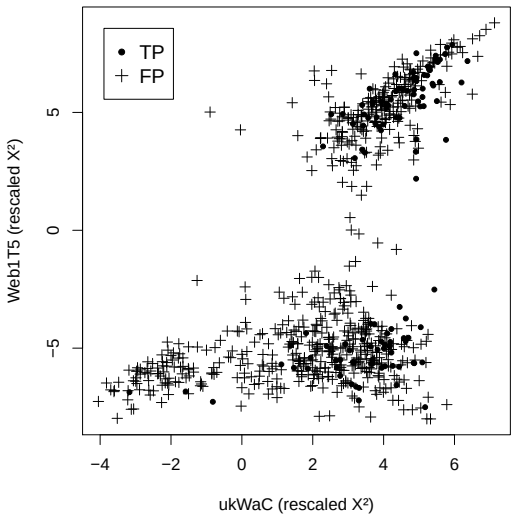


Figure 2: Comparison of X^2 association scores on ukWaC and Web1T5. Axes are rescaled logarithmically, preserving sign to indicate positive vs. negative association.

In order to determine whether the poor performance of Web1T5 is simply due to the lack of linguistic annotation or whether it points to an intrinsic problem of the n-gram database, co-occurrence data were extracted from an untagged version of the BNC using the same method as for the Web1T5 data. While there is a significant decrease in precision (cf. Figure 3.g and the last row of Table 4), the results are still considerably better than on Web1T5. In the MWE 2008 competition, Ramisch et al. (2008) were also unable to improve on the BNC results using a phrase entropy measure based on search engine data.

The direct comparison of X^2 association scores on ukWaC and Web1T5 in Figure 2 reveals that the latter are divided into strongly positive and strongly negative association, while scores on ukWaC are spread evenly across the entire range. It is remarkable that many true positives (TP) exhibit negative association in Web1T5, while all but a few show the expected positive association in ukWaC. This unusual pattern, which may well explain the poor VPC evaluation results, can also be observed for adjacent bigrams extracted from the 2-grams table (not shown). It suggests a general problem of the Web1T5 data that is compounded by the quasi-collocations approach.

6 Future work

A new release of Web1T5-Easy is currently in preparation. It will refactor the Perl code into reusable and customisable modules that can easily be embedded in user scripts and adapted to other databases such as Brants and Franz (2009). We are looking forward to Web1T5 v2, which promises easier indexing and much richer interactive queries.

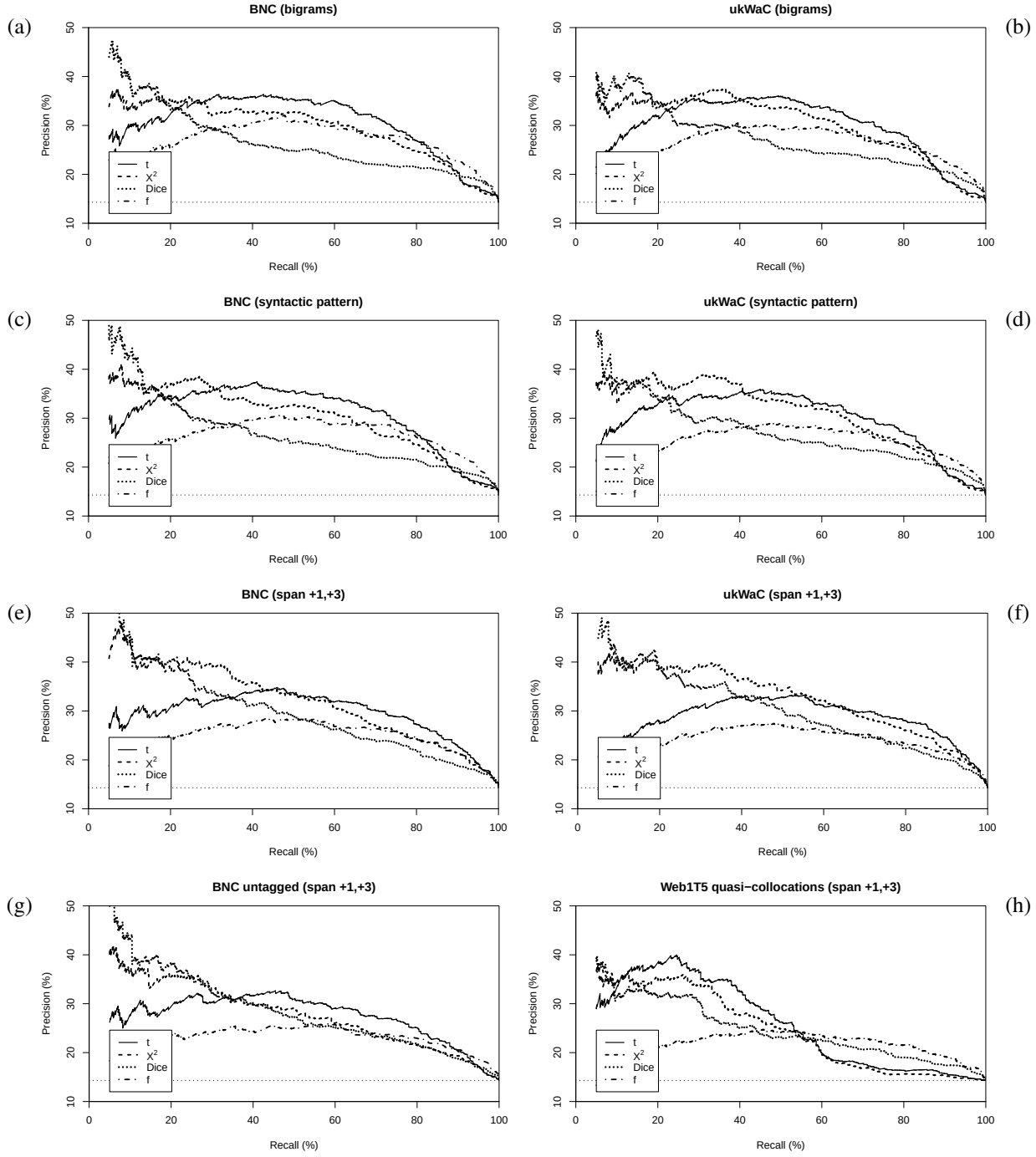


Figure 3: Evaluation results for English non-compositional VPC (Baldwin, 2008): precision-recall graphs. Rankings according to the Web1T5 quasi-collocations are shown in the bottom right panel (h). The baseline precision is 14.29%.

References

Guy Aston and Lou Burnard. 1998. *The BNC Handbook*. Edinburgh University Press, Edinburgh. See also the BNC homepage at `http://www.natcorp.ox.ac.uk/`.

Timothy Baldwin. 2008. A resource for evaluating the deep lexical acquisition of English verb-particle constructions. In *Proceedings of the LREC Workshop Towards a Shared Task for Multiword Expressions (MWE 2008)*, pages 1–2, Marrakech, Morocco.

Marco Baroni, Silvia Bernardini, Adriano Ferraresi, and Eros Zanchetta. 2009. The WaCky Wide Web: A collection of very large linguistically processed Web-crawled corpora. *Language Resources and Evaluation*, 43(3):209–226.

Thorsten Brants and Alex Franz. 2006. *Web 1T 5-gram Version 1*. Linguistic Data Consortium, Philadelphia, PA. `http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2006T13`.

Thorsten Brants and Alex Franz. 2009. *Web 1T 5-gram, 10 European Languages Version 1*. Linguistic Data Consortium, Philadelphia, PA. `http://www.ldc.upenn.edu/Catalog/CatalogEntry.jsp?catalogId=LDC2009T25`.

James Richard Curran. 2004. *From Distributional to Semantic Similarity*. Ph.D. thesis, University of Edinburgh.

Stefan Evert. 2008. Corpora and collocations. In Anke Lüdeling and Merja Kytö, editors, *Corpus Linguistics. An International Handbook*, chapter 58. Mouton de Gruyter, Berlin.

Tobias Hawker, Mary Gardiner, and Andrew Bennetts. 2007. Practical queries of a massive n-gram database. In *Proceedings of the Australasian Language Technology Workshop 2007*, pages 40–48, Melbourne, Australia.

Tom M. Mitchell, Svetlana V. Shinkareva, Andrew Carlson, Kai-Min Chang, Vicente L. Malave, Robert A. Mason, and Marcel Adam Just. 2008. Predicting human brain activity associated with the meanings of nouns. *Science*, 320:1191–1195.

Carlos Ramisch, Paulo Schreiner, Marco Idiart, and Aline Villavicencio. 2008. An evaluation of methods for the extraction of multiword expressions. In *Proceedings of the LREC Workshop Towards a Shared Task for Multiword Expressions (MWE 2008)*, pages 50–53, Marrakech, Morocco.

John Sinclair. 1991. *Corpus, Concordance, Collocation*. Oxford University Press, Oxford.