The Effect of Syntactic Phrase Indexing on Retrieval Performance for Dutch Texts

Renée Pohlmann
UiL OTS, Utrecht University
Trans 10
3512 JK Utrecht
The Netherlands
tel. +31-302536064
fax. +31-302536000
Renee.C.Pohlmann@let.ruu.nl

Wessel Kraaij
TNO-TPD
Postbus 155
6200 AD Delft
The Netherlands
tel. +31-152692259
fax. +31-152692000
kraaij@tpd.tno.nl

Abstract

In this paper we describe an experiment with syntactic phrase indexing for Dutch texts. We compare different choices for combining terms to form head-modifier pairs and we also investigate the effect of adding none, one, or all constituent parts of the pair as a separate index term. The results of our experiments show that using head-modifier pairs as index terms can improve both recall and precision significantly but only if all constituent parts are also added separately. We found that using both Noun-Adjective and Noun-Noun head-modifier pairs produced the best results.

Keywords

Natural language processing; syntactic phrase indexing; head-modifier pairs; Dutch.

1 Introduction

The work described in this paper is part of the UPLIFT project[1]. UPLIFT investigates whether linguistic tools can improve and extend the functionality of vector space text retrieval systems. This paper describes an experiment with syntactic phrase indexing for Dutch texts. The basic idea behind phrase indexing is that phrases characterize document content more effectively than single word terms. When using a single word index, a query containing the phrase "information retrieval" will also match with documents containing only "information" or "retrieval". If "information retrieval" is recognized as a unit, however, these matches may be avoided or given a much lower score (depending on the matching strategy). Researchers have used different strategies to identify suitable phrases for indexing, the most important distinction being between strategies based on statistical co-occurence data and strategies based on syntactic processing. So far, both types of strategies have proven to be equally succesful (cf. e.g. [Fag87] and [SBS90]). Our approach was inspired by the work of Strzalkowski, as described in [Str95] and [SPC96]. Other recent work on syntactic phrase indexing includes [EZ96] and [SOK95]. Strzalkowski uses syntactic information to identify phrases in queries and documents. These phrases are subsequently "normalized" (i.e. semantically similar but syntactically different contructions, e.g. retrieval of information vs. information retrieval, are represented identically) as head-modifier pairs. In the next section (2) we will describe our experiments and present the results and in section 3 we will summarize the main conclusions and outline our future research.

2 Experiments

2.1 System variants

Earlier research in the UPLIFT project showed that when a query is expanded with the constituents of compounds already occurring in it[2] and new compounds are added to the query by combining query terms[3], recall improves while precision does not deteriorate. The following example illustrates this approach.

Query: *Ik zoek documenten over computers en natuurlijke taalverwerking* ("I am looking for documents on computers and natural language processing")

This query would result in the following index terms (after removal of stop words):

document	
computer	
natuurlijk	
taalverwerking	
taal	compound splitting
verwerking	"
computertaal	compound generation
taalcomputer	"

[1] UPLIFT: Utrecht Project: Linguistic Information for Free Text retrieval. UPLIFT Home page: *http://wwwots.let.ruu.nl/~uplift*

[2] In Dutch, compounds are usually written as a single orthographic unit, e.g. levensverzekeringsmaatschappij (life insurance company).

[3] Newly created compounds are verified against a database of compounds collected from the documents.

In the example, the compounds “computertaal” (computer language) and “taalcomputer” (language computer) are added to the query by combining “computer” and “taal”. Both are valid compounds but, although the second compound may retrieve relevant articles for this query, the first (a synonym for programming language) will probably retrieve many unrelated documents.

We decided to investigate whether it would be possible to improve this procedure by using syntactic information to constrain the compound splitting and compound generation processes. We investigated whether compound heads (“verwerking” in the example above) are better index terms than modifiers (“taal” in the example above). Compound generation was restricted to Noun-Noun head-modifier term pairs originating from complex nominal compounds (i.e. consisting of more than two constituents) and Noun Phrases (NPs) containing a specific type of Prepositional Phrase (PP) as a noun modifier. The choice for the latter type of construction was motivated by the fact that many compounds in Dutch can be paraphrased using a PP, e.g. *fietswiel* (“bicycle wheel”) $\leftrightarrow$ *wiel van een fiets* (“wheel of a bicycle”), see, for instance, [GHdRvdT84] p. 103.

We subsequently developed a term pair identification and extraction module and integrated it with our basic retrieval engine[4]. This module consists of the following parts. First, a segmentation algorithm is used to identify sentence and word boundaries in the texts. A lexical lookup algorithm, based on the CELEX electronic dictionary for Dutch [BPvR93], assigns lexical tags to the words. A tagger is used to resolve ambiguities in tag assignment. We used the Multext tagger [ABR95], a Hidden Markov Model tagger, which has the advantage that it requires only a partially disambiguated corpus for training. After training, the tagger produced 91.5% correct results[5]. Next, a very simple heuristic based on the distinction upper case–lower case is used to glue sequences of proper names together (e.g. Verenigde_Staten_van_Amerika (United States of America)). We use an NP-parser to parse and extract NPs from the texts. The parser was developed by TNO-TPD [vSdB93]. This parser is deterministic and requires fully disambiguated input from the tagger but it is robust and fast (244 sentences per second on a Sun-Sparc 10/40). The coverage of the grammar is not complete[6] but for the purpose of our experiment, identifying complex NPs with PP-modifiers, it was considered to be sufficient. A separate term pair extraction module then extracts the head-modifier pairs from the output of the parser. Identifying head-modifier relationships in compounds is not trivial because of possible structural ambiguities. In Dutch, compounds existing of two parts are usually right-headed (a “fietswiel” is a sort of “wiel”) but compound construction is recursive and both the head and the modifier can be compounds themselves resulting in structural ambiguities, e.g. [[X1 X2] X3] or [X1 [X2 X3]]. To split up compounds into their constituent parts we use the lexicon-based compound splitter developed by Theo Vosse for the CORRie project [Vos94]. We have not attempted to implement a strategy to solve all structural ambiguities in compounds but we have applied two[7] different heuristics to extract probable pairs. In a recent study [tS96], ter Stal found that simply assuming that all compounds have a left-branching structure produced $\pm$ 70% correct results. Although his results are for English we decided to try this strategy for our pair extraction routine. As an alternative, we also implemented a strategy where we use unambiguous cases collected from the

[4] The retrieval engine used in the UPLIFT project is the TRU vector space engine developed by Philips Research [ABC91].

[5] As better results for Dutch have been reported in the literature for other taggers (cf. [DK95]) it would be interesting to see if retrieval performance would improve by using a different tagger.

[6] Relative clauses, for instance, are not included.

[7] A third option where the structure is simply left ambiguous and all interpretations are selected, was not implemented for lack of time. We intend to test this version is the future.

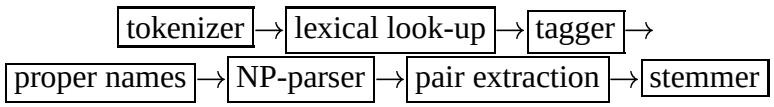

Figure 1: term pair identification and extraction module

corpus to confirm a certain choice. If we find independent evidence for a left-branching structure (X1 modifies X2 in unambiguous contexts) or a right-branching structure (X1 modifies X3) we select the pairs accordingly. If we do not find independent evidence for either structure we choose a left-branching structure by default. PP-modification structures have similar ambiguities, e.g. in "the man with the dog with the spots" it is not clear whether the PP "with the spots" modifies "the man" or "the dog". These structures are treated analogously to the compound structures[8]. Finally, all words (and compound constituents) are replaced by their stem using a lexicon-based (CELEX) stemming algorithm[9]. Figure 1 illustrates the term pair identification and extraction process.

Based on the different options described above we developed and tested 17 new system variants. The versions can be divided into the following classes.

vn Basic engine, no extensions.

vXXX Basic version extended with tagging, proper name identification and stemming.

vM.. Noun-Noun head-modifier pairs from complex compounds are added to the index.

vS.. Both Noun-Noun head-modifier pairs from compounds and PP-constructions (compound paraphrases) are added to the index.

v.a. Pairs are selected using a default strategy (i.e. for ambiguous compounds a left-branching structure is chosen and for ambiguous PP-constructions a right-branching structure).

v.c. Independent evidence from the corpus is used in pair selection.

v..1 Constituents of head-modifier pairs are also added separately to the index.

v..2 Only heads (including heads of complex modifiers) are added to the index separately.

v..3 Only the head of the entire construction is added as a separate index term.

v..4 Pair constituents are not added to the index.

c4fow Best version from previous experiments, all subparts of compounds are added to the query and new compounds are generated by arbitrarily combining query terms.

[8] Note that because of the reversed order of head and modifiers we choose a right-branching structure by default in these cases.

[9] We used the best variant of all the stemming algorithms tested in previous UPLIFT experiments (cf. [KP96b]). This variant handles inflection only.

2.2 Results

The results of the experiment are summarized in table 1. The test collection used for the experiments was compiled during previous research in the UPLIFT project on stemming algorithms. It consists of approximately 60.000 Dutch newspaper articles and 36 queries and relevance judgements. For a more detailed description of the test collection and test procedures we refer to [KP96a]. We used 4 different evaluation measures, which we considered representative, to evaluate retrieval performance. These evaluation measures are: average precision, ap5-15 (average precision at 5, 10 and 15 documents retrieved), R-recall (recall at R, where R is the number of relevant articles for a particular query[10]) and recall1000 (recall at 1000 documents retrieved). We also performed statistical significance tests to establish whether the differences between means of evaluation measures are significant or should be attributed to chance. The design chosen for these statistical tests is based on [TS95a] and [TS95b]. The results of the statistical tests can be found in in the appendix.

Compared to the reference version (vn)[11], significantly better results have been achieved for 3 out of 4 evaluation measures. Tagging, proper name identification and stemming alone (version vXXX) already improve retrieval performance by up to 12%. The results show that the distinction head-modifier is not relevant for compound splitting. Only if all subparts of a head-modifier pair are also added to the index (versions v..1), do we achieve positive results. Although versions vM.2 and vM.3 do show a slight advantage over vM.1 for ap5-15, this difference is not statistically significant. We have not been able to show a difference between the two strategies for handling ambiguous structures (v.a. and v.c.); both strategies perform comparably. It may be that our corpus is too small to render sufficient data for the corpus-based approach. It may also be that the default strategy simply works well for our data. Using syntactic information to guide the compound formation process (versions vM.. and vS..) results in a significantly better performance for 3 out of 4 evaluation measures. Compared to vn, only average precision has not improved. We may conclude that adding head-modifier pairs to the index improves retrieval performance, but only if all constituent parts are also added as separate index terms. Although this is standard practice, we know of no other research which has systematically addressed this issue. The difference between the new syntactic versions and c4fow, however, is not significant for any of the evaluation measures. In table 2 some statistics for versions c4fow and vSa1 are given. The figures show that although the number of compounds found by c4fow greatly exceeds the number of compounds found by the syntactic version, the percentage of relevant combinations (actually found in relevant articles) is higher for the syntactic version. It may be that the compound generation strategy employed by the syntactic versions is too restricted and should be extended to include other head-modifier pairs. We experimented with several extensions. The results of these experiments will be given in section 2.3 below.

2.3 Further experimentation

The results described above inspired us to experiment further and develop other versions of the system using different syntactic restrictions. We implemented a version which instead of a subset of PP-modifiers (those considered compound paraphrases, cf. section 2.1 above) uses

[10] For a motivation for this particular evaluation measure see [KP96b], p.44.

[11] The results for c4fow and vn are slightly different from those reported in [KP96b]. This is a result of using a different interpolation method (probability of relevance, cf. [RJ89]) for calculating recall/precision values.

version	avp	% change		ap5-15	% change	
vMa1	**0.34972** (0.21535)	+	**11.86**	0.44290 (0.28363)	+	14.25
vMa2	0.33997 (0.22486)	+	8.74	0.44815 (0.30933)	+	15.61
vMa3	0.34374 (0.22668)	+	9.95	**0.45093** (0.31084)	+	**16.32**
vMa4	0.32962 (0.21242)	+	5.43	0.42654 (0.27442)	+	10.03
vMc1	0.34955 (0.21536)	+	11.81	0.44352 (0.28373)	+	14.41
vMc2	0.34087 (0.22491)	+	9.03	0.44568 (0.30846)	+	14.97
vMc3	0.34356 (0.22689)	+	9.89	0.45031 (0.31074)	+	16.16
vMc4	0.33025 (0.21219)	+	5.63	0.42593 (0.27427)	+	9.87
vSa1	0.34596 (0.21406)	+	10.66	0.44136 (0.28298)	+	13.86
vSa2	0.28359 (0.24023)	-	9.29	0.36574 (0.33411)	-	5.65
vSa3	0.28480 (0.23983)	-	8.90	0.36080 (0.33457)	-	6.93
vSa4	0.27743 (0.23568)	-	12.26	0.34907 (0.32906)	-	9.95
vSc1	0.34807 (0.21312)	+	11.33	0.44321 (0.28244)	+	14.33
vSc2	0.28571 (0.23937)	-	8.61	0.36512 (0.33113)	-	5.81
vSc3	0.28569 (0.23845)	-	8.62	0.36327 (0.33104)	-	6.29
vSc4	0.27882 (0.23516)	-	10.82	0.34907 (0.32906)	-	9.95
vXXX	0.32963 (0.21809)	+	5.43	0.42037 (0.28499)	+	8.44
c4fow	0.31946 (0.20027)	+	2.18	0.42654 (0.27684)	+	10.03
vn	0.31264 (0.21436)			0.38765 (0.29079)		
version	R-recall	% change		recall1000	% change	
vMa1	**0.34369** (0.18631)	+	**22.11**	0.91417 (0.10226)	+	19.46
vMa2	0.31305 (0.20627)	+	11.22	0.87880 (0.13591)	+	14.84
vMa3	0.31243 (0.20703)	+	11.00	0.87532 (0.14068)	+	14.39
vMa4	0.31056 (0.20148)	+	10.34	0.85022 (0.16538)	+	11.10
vMc1	0.34332 (0.18627)	+	21.97	0.91417 (0.10225)	+	19.46
vMc2	0.31185 (0.20670)	+	10.79	0.87294 (0.14504)	+	14.07
vMc3	0.31243 (0.20703)	+	11.00	0.87493 (0.14129)	+	14.33
vMc4	0.31043 (0.20135)	+	10.29	0.84873 (0.16721)	+	10.91
vSa1	0.34254 (0.18420)	+	21.70	0.91760 (0.10399)	+	19.91
vSa2	0.25982 (0.21201)	-	7.69	0.78350 (0.22996)	+	2.38
vSa3	0.25440 (0.21601)	-	9.62	0.75842 (0.24561)	-	0.89
vSa4	0.24620 (0.21153)	-	12.53	0.74629 (0.24910)	-	2.48
vSc1	0.34273 (0.18414)	+	21.76	**0.91766** (0.10397)	+	**19.92**
vSc2	0.26020 (0.21023)	-	7.56	0.77803 (0.23177)	+	1.67
vSc3	0.25571 (0.21384)	-	9.15	0.76019 (0.24141)	-	0.66
vSc4	0.24816 (0.21034)	-	11.83	0.74551 (0.24718)	-	2.58
vXXX	0.31566 (0.20602)	+	12.15	0.85539 (0.16553)	+	11.78
c4fow	0.31727 (0.19130)	+	12.72	0.88075 0.14841	+	15.09
vn	0.28147 (0.19464)			0.76525 (0.21618)		

Table 1: Evaluation measures averaged over queries (including variance)

version	number of compounds	relevant compounds	% relevant
c4fow	147	35	20.47
vSa1	46	18	39.13

Table 2: relevant compounds found by c4fow vs. vSa1

all PP-modifiers for term pair generation (version vP1). Besides this version we also developed a version which adds Noun-Adjective head-modifier pairs to the index (vA1). Version vAP1 combines these two strategies. The results for these versions are given in table 3. The results for c4fow and vn are repeated here for ease of reference.

version	avp	% change	ap5-15	% change
vA1	0.34722 (0.20990)	+ 11.06	0.44290 (0.27705)	+ 14.25
vAP1	**0.35814** (0.21702)	+ **14.55**	0.44846 (0.27628)	+ 15.69
vP1	0.35401 (0.22111)	+ 13.23	**0.45062** (0.28228)	+ **16.24**
c4fow	0.31946 (0.20027)	+ 2.18	0.42654 (0.27684)	+ 10.03
vn	0.31264 (0.21436)		0.38765 (0.29079)	
version	R-recall	% change	recall1000	% change
vA1	0.34285 (0.18762)	+ 21.81	**0.92000** (0.09959)	+ **20.22**
vAP1	**0.35413** (0.19475)	+ **25.81**	0.91780 (0.10453)	+ 19.93
vP1	0.34792 (0.18856)	+ 23.61	0.91946 (0.10013)	+ 20.15
c4fow	0.31727 (0.19130)	+ 12.72	0.88075 0.14841	+ 15.09
vn	0.28147 (0.19464)		0.76525 (0.21618)	

Table 3: Evaluation measure averaged over queries (including variance)

Compared to the reference version we have now achieved a significantlty better performance for all 4 evaluation measures, including average precision. vAP1, the version which adds both Noun-Adjective and Noun-Noun pairs being the best overall version. Although c4fow, the version which does not use any syntactic information, performs surprisingly well (up to 15% improvement compared to vn), we are able to improve results even further (up to 25% improvement compared to vn) by adding syntactic information. The data also suggest that it might be interesting to investigate what the effect might be of adding even more head-modifier pairs (originating from relative clause modification, for instance) to the index.

3 Conclusions and future work

The results of our experiments have shown that it is possible to improve retrieval quality for Dutch texts significantly by using syntactic information. Adding term pairs to the index can improve retrieval performance by up to 25%, provided that all subparts are also added to the index separately. For the expriments described above we used a *tf.idf* weighting scheme which does not differentiate between simple and complex index terms. Since term re-weighting schemes have proven to be successful in previous UPLIFT experiments, we intend to investigate the effect of alternative weighting strategies in the future. We also plan to adapt our strategy to English texts and investigate cross-language retrieval.

Acknowledgements

The research described in this paper was funded by Philips Research, TNO-TPD, the Utrecht Institute of Linguistics (OTS), the NBBI, the Dutch Ministry of Education and Science and the Dutch Ministry of Economics. The authors would like to thank Herbert Ruessink and Dirk Heylen for their contributions to this work.

References

[ABC91] IJsbrand Jan Aalbersberg, Ewout Brandsma, and Marc Corthout. Full text document retrieval: from theory to applications. *Informatiewetenschap 1991, Wetenschappelijke bijdragen aan de eerste STINFON-Conferentie*, 1991.

[ABR95] Susan Armstrong, Pierrette Bouillon, and Gilbert Robert. Tools for part-of-speech tagging. Multext project report, ISSCO, Geneva, 1995.

[BPvR93] R. H. Baayen, R. Piepenbrock, and H. van Rijn, editors. *The CELEX Lexical Database (CD-ROM)*. Linguistic Data Consortium, University of Pennsylvania, Philadelphia (PA), 1993.

[DK95] Evangelos Dermatas and George Kokkinakis. Automatic stochastic tagging of natural language texts. *Computational Linguistics*, 21(2):137–163, 1995.

[EZ96] David A. Evans and Chengxiang Zhai. Noun-phrase analysis in unrestricted text for information retrieval. In *Proceedings ACL96*, 1996.

[Fag87] J. L. Fagan. *Experiments in Automatic Phrase Indexing for Document Retrieval: A Comparison of Syntactic and non-Syntactic Methods*. PhD thesis, Cornell University, Ithaca NY, CS Department technical report 87-868, 1987.

[GHdRvdT84] G. Geerts, W. Haeseryn, J. de Rooij, and M. C. van der Toorn, editors. *Algemene Nederlandse Spraakkunst*. Wolters Noordhoff, Groningen, 1984.

[KP96a] Wessel Kraaij and Renée Pohlmann. Using linguistic knowledge in information retrieval. OTS Working Paper OTS-WP-CL-96-001, Research Institute for Language and Speech (OTS), Utrecht University, 1996.

[KP96b] Wessel Kraaij and Renée Pohlmann. Viewing stemming as recall enhancement. In Hans-Peter Frei, Donna Harman, Peter Schauble, and Ross Wilkinson, editors, *Proceedings of ACM-SIGIR96*, pages 40–48, 1996.

[RJ89] Vijay V. Raghavan and Gwang S. Jung. A critical investigation of recall and precision as measures of retrieval system performance. *ACM Transactions on Information Systems*, 7(3):205–229, 1989.

[SBS90] Gerard Salton, Chris Buckley, and Maria Smith. On the application of syntactic methodologies in automatic text analysis. *Information Processing & Management*, 26(1):73–92, 1990.

[SOK95] Alan F. Smeaton, Ruairi O'Donnell, and Fergus Kelledy. Indexing structures derived from syntax in TREC-3: System description. In Donna Harman, editor, *Overview of The Third Text REtrieval Conference (TREC-3)*, pages 55–63. National Institute for Standards and Technology, 1995. Special Publication 500-225.

[SPC96] Tomek Strzalkowski and Jose Perez Carballo. Natural language information retrieval: TREC-4 report. In Donna Harman, editor, *The Fourth Text REtrieval Conference (TREC-4)*. National Institute for Standards and Technology, 1996. Special Publication 500-236.

[Str95] Tomek Strzalkowski. Natural language information retrieval. *Information Processing & Management*, 31(3):397–417, 1995.

[TS95a] Jean Tague-Sutcliffe. *Measuring Information, An Information Services Perspective*. Academic Press, San Diego (CA), 1995.

[TS95b] Jean Tague-Sutcliffe. A statistical analysis of the TREC-3 data. In Donna Harman, editor, *Overview of the Third Text REtrieval Conference (TREC-3)*, pages 385–398. National Institute for Standards and Technology, 1995. Special Publication 500-225.

[tS96] Wilco ter Stal. *Automated Interpretation of Nominal Compounds in a Technical Domain*. PhD thesis, Technische Universiteit Twente, UT Repro, Enschede, 1996.

[Vos94] T. G. Vosse. *The Word Connection*. PhD thesis, Rijksuniversiteit Leiden, Neslia Paniculata Uitgeverij, Enschede, 1994.

[vSdB93] Joost van Surksum and Jan Willem den Besten. Patz'er - een patronenzoeker voor Nederlandstalige teksten. TNO-TPD/Hogeschool Enschede, november 1993.

Appendix: results of the statistical analysis

Tables 4, 5, 6 and 7 present the results of the analysis of variance that was run on the data.

Source	DF	Sum of Sq	Mean Sq	F val
system	21	0.6067	0.0289	4.1143
queries	35	34.1529	0.9758	138.9643
error	735	5.1611	0.0070	
total	791	39.9207	0.0505	
sed (systems)	0.0198			

Table 4: ANOVA table average precision

Source	DF	Sum of Sq	Mean Sq	F val
system	21	1.1230	0.0535	4.6616
queries	35	63.2049	1.8059	157.4243
error	735	8.4314	0.0115	
total	791	72.7592	0.0920	
sed (systems)	0.0252			

Table 5: ANOVA table average precision at 5, 10 and 15 documents retrieved

Source	DF	Sum of Sq	Mean Sq	F val
system	21	1.0188	0.0485	6.4021
queries	35	26.3084	0.7517	99.1905
error	735	5.5699	0.0076	
total	791	32.8971	0.0416	
sed (systems)	0.0205			

Table 6: ANOVA table R-recall

The most important figures in the ANOVA tables are the F-values in the rightmost column, which represent the quotient of the variance in measurements which can be attributed to the effect we are interested in and the variance due to chance. This quotient is dependent on the degrees of freedom of the variables in the model i.e. number of system versions and queries. The F distribution shows us that the system effect is significant at the 0.99 level for all ANOVAS, because the F values for system effect exceed $F_{.99;21,735}$[12] = 1.85. This means that we can reject the hypotheses that the system effects of the corresponding measures are equal to zero with a certainty of 99%. The query effect (query column) is also clearly significant: the F-values exceed $F_{.99;35,735}$ = 1.55.

Because the ANOVA shows that there are significant differences between system versions, it is necessary to do multiple pairwise comparisons to detect which specific versions are concerned. We have used T-tests to identify significant differences between specific versions.

The SED values are used to discriminate significantly different versions in the following way:

(1) $$\mid \bar{x}_1 - \bar{x}_2 \mid > 2 \times s.e.d.$$

[12]The subscripts refer to the significance level (1-0.01) and the degrees of freedom.

Source	DF	Sum of Sq	Mean Sq	F val
system	21	3.2475	0.1546	11.7376
queries	35	13.8056	0.3944	29.9390
error	735	9.6837	0.0132	
total	791	26.7368	0.0338	
sed (systems)	0.0271			

Table 7: ANOVA table recall at 1000 documents

Tables 8, 9, 10 and 11 present the results of the multiple comparisons. The diagrams must be interpreted as follows: if two means are connected by the same line segment, their difference is not significant.

system	avp
vAP1	0.35814
vP1	0.35401
vMa1	0.34972
vMc1	0.34955
vSc1	0.34807
vA1	0.34722
vSa1	0.34596
vMa3	0.34374
vMc3	0.34356
vMc2	0.34087
vMa2	0.33997
vMc4	0.33025
vXXX	0.32963
vMa4	0.32962
c4fow	0.31946
vn	0.31264
vSc2	0.28571
vSc3	0.28569
vSa3	0.28480
vSa2	0.28359
vSc4	0.27882
vSa4	0.27743

Table 8: T-tests avp

system	ap5-15
vMa3	0.45093
vP1	0.45062
vMc3	0.45031
vAP1	0.44846
vMa2	0.44815
vMc2	0.44568
vMc1	0.44352
vSc1	0.44321
vMa1	0.44290
vA1	0.44290
vSa1	0.44136
vMa4	0.42654
c4fow	0.42654
vMc4	0.42593
vXXX	0.42037
vn	0.38765
vSa2	0.36574
vSc2	0.36512
vSc3	0.36327
vSa3	0.36080
vSa4	0.34907
vSc4	0.34907

Table 9: T-tests ap5-15

system	R-recall
vAP1	0.35413
vP1	0.34792
vMa1	0.34369
vMc1	0.34332
vA1	0.34285
vSc1	0.34273
vSa1	0.34254
c4fow	0.31727
vXXX	0.31566
vMa2	0.31305
vMa3	0.31243
vMc3	0.31243
vMc2	0.31185
vMa4	0.31056
vMc4	0.31043
vn	0.28147
vSc2	0.26020
vSa2	0.25982
vSc3	0.25571
vSa3	0.25440
vSc4	0.24816
vSa4	0.24620

Table 10: T-tests R-recall

system	recall1000
vA1	0.92000
vP1	0.91946
vAP1	0.91780
vSc1	0.91766
vSa1	0.91760
vMc1	0.91417
vMa1	0.91417
c4fow	0.88075
vMa2	0.87880
vMa3	0.87532
vMc3	0.87493
vMc2	0.87294
vXXX	0.85539
vMa4	0.85022
vMc4	0.84873
vSa2	0.78350
vSc2	0.77803
vn	0.76525
vSc3	0.76019
vSa3	0.75842
vSa4	0.74629
vSc4	0.74551

Table 11: T-tests recall1000