The University of Amsterdam at TREC 2004

David Ahn Valentin Jijkoun Jaap Kamps* Gilad Mishne
Karin Müller Maarten de Rijke Stefan Schlobach†

Informatics Institute, University of Amsterdam
Kruislaan 403, 1098 SJ Amsterdam, The Netherlands
http://ilps.science.uva.nl/

Abstract: We describe our participation in the TREC 2004 Web, Terabyte, and Question Answering tracks. We provide a detailed account of the ideas underlying our approaches to these tasks, report on our results, and give a summary of our findings so far.

1 Introduction

At TREC 2004 we took part in the Web, Terabyte, and Question Answering tracks. Our aim for the Web track was to investigate a range of web-centric retrieval techniques based on an analysis of non-content features, such as document length, URL structure, and link topology. Our aim for the Terabyte track was to set-up an initial system based on compact document representations such as titles or incoming anchor texts, and to compare the relative effectiveness of these document surrogates. Our aim for the Question Answering track was to extend our QA system to handle this year's more complex question presentation, and to see how our existing modules cope with this new setting.

The rest of this paper is organized as follows. In three largely self-contained sections we describe our work for the Web (§2), Terabyte (§3), and Question Answering (§4) tracks. We summarize our findings in a concluding section.

*Currently at Archives and Information Studies, Faculty of Humanities, University of Amsterdam.

†Currently at the Division of Mathematics and Computer Science, Free University of Amsterdam.

2 Web Track

We experimented with a range of techniques within the language modeling framework, exploiting natural ways to incorporate multiple document representations, as well as non-content information. We use three indexes based on document-text, incoming anchor-texts, and document titles, similar to those used for our submissions to TREC 2003 [8].

2.1 Mixture Language Models

For the web tasks we use a specific mixture language model based on the following formula:

$$P(q|d) = P(d) \cdot \prod_{i=1}^{n} ((1-\lambda) \cdot P(q_i|C) + \lambda \cdot P(q_i|d)).$$

For the web track we have three document models:

1. $P_{\text{text}}(q_i|d)$ the estimate based on the full-text index.
2. $P_{\text{anchor}}(q_i|d)$ the estimate based on the anchortext index.
3. $P_{\text{title}}(q_i|d)$ the estimate based on the titles index.

This leads to the formula:

$$P(q|d) = P(d) \cdot \prod_{i=1}^{n} ((1-\lambda_1-\lambda_2-\lambda_3) \cdot P(q_i|C) + \lambda_1 \cdot P_{\text{text}}(q_i|d) + \lambda_2 \cdot P_{\text{anchor}}(q_i|d) + \lambda_3 \cdot P_{\text{title}}(q_i|d)),$$

where each of the document models is estimated using a maximum likelihood estimate. All runs on which report

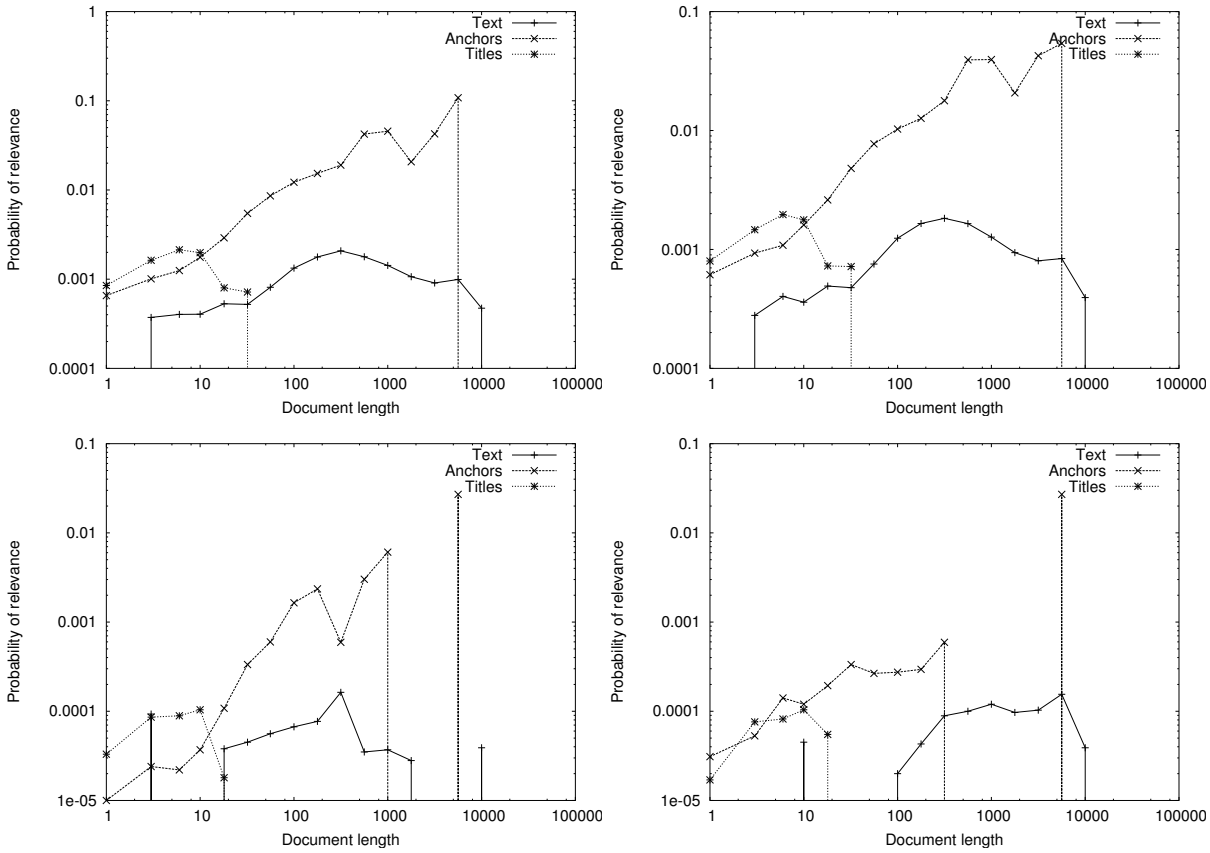

Figure 1: Document length versus relevance overall (top left), and for distillation (top right), home page (bottom left), and named page topics (bottom right).

below use equal weights for all three document models, that is $\lambda_1 = \lambda_2 = \lambda_3 = \lambda$.

We use the full text index as the collection model. The prior probability of a document, $P(d)$, can be used to incorporate non-content features into the scoring mechanism, as we will now explain.

2.2 Priors

We will now analyze the a range of non-content features, such as the document length, the page's URL, or the link topology, and investigate their usefulness to boost retrieval effectiveness.

2.2.1 Document length

Let us focus on document length first. Figure 1 shows the prior probability of relevance against the length of a document for each of the three indexes (full-text, anchor-texts, and titles). The plot at the top left of the figure shows the prior probability of relevance of a web page for any of the mixed query topics. If we consider all mixed query topics, plotted in the figure at the top left, then the only marked length effect is for the anchor-text index.

Even though the three topic types are evenly distributed, the number of relevant pages is not. Table 1 shows the number of relevant pages for each of the topic types in the TREC 2004 qrels. So, for over 90 percent the

Table 1: Number of relevant pages per topic type.

Type	Topics	# Rel	% Rel	Rel/Top
Topic distillation	75	1,600	90.8%	21.33
Home pages	75	83	4.7%	1.11
Named pages	75	80	4.5%	1.07
Mixed queries	225	1,763	100%	7.84

observed patterns can be attributed to the distillation topics. This is confirmed by looking at the results for the distillation topics only (top right plot in the Figure 1). As it turns out, for the other subtasks, home page finding (bottom left plot) and named page finding (bottom right plot), the results are fairly similar: the only marked length effect can be observed for the anchor-text index.

For each of the tasks the relevance of a page seems unrelated to the length of the page. It does have a relation with the length of a document in the anchor-text index. The length of the anchor-text document surrogate is directly correlating with the number of incoming links. Since the indegree of a page provides a more direct handle, we decided not to use document length as a factor for our web retrieval experiments.

2.2.2 URL

We will now focus on the uniform resource locator (URL) as a non-content feature, independent of the particular query at hand. Table 2 shows the prior probability of relevance for the familiar URL classes [9]. Note that, again,

Table 2: Priors for the URL classes.

Class	Mixed	TD	HP	NP
Root	0.046845	0.042559	0.003990	0.000296
Subroot	0.003225	0.002894	0.000215	0.000116
Path	0.003440	0.003183	0.000167	0.000091
File	0.000786	0.000713	0.000018	0.000055

the results for the mixed queries are dominated by the distillation topics since they populate the pool of relevant documents. We break down the set of topics for the three individual topic types. The results for home page finding and named page finding are only in partial agreement with the distillation topics. There is a reversal of the relative importance for the Subroot and Path classes for the known-item topics. Also, for the named page topics, the Root class pages are only moderately more relevant, on average, than pages in the Subroot class. Although it is clear that these coarse-grained URL classes can be used as a prior for retrieval, we want to investigate more fine-grained measures of URL length.

We first normalize the URLs by removing "www" prefixes, and "index.htm(l)" postfixes. We investigate three measures of the length of the URL:

URL Slash Count Simply count the number of occurrences of "/" in the URL. For example `trec.nist.gov/act_part/act_part.html` has a slash count of 2.

URL Character Length Simply count the number of symbols in the URL. For example `trec.nist.gov/act_part/act_part.html` has a character length 36.

URL Component Length Split the URL in the *domain name* and *file path*, count the number of "." separated components in the domain name, and count the number of "/" separated components in the file path. For example `trec.nist.gov/act_part/act_part.html` will split in the domain name `trec.nist.gov` and the file path `act_part/act_part.html`. The domain name has 3 components, and the file path 2, making a component length of 5.

Figure 2 shows the prior probability of relevance for the three measures of URL length. The length of a URL has a clear reciprocal relation with relevancy: the shorter the URL, the more likely the page is to be relevant. Although all three URL length indicators can be used, pre-submission experiments on TREC 2003 data suggested that URL component length is the most promising.

In particular, we experimented with three operationalizations to the URL priors:

Linear The prior is proportional to $11 - component\_length$ if the length is maximally 10, use 0.1 otherwise.

Linear Squared The prior is proportional to the square of the linear prior.

Product The prior is proportional to $\frac{1}{component\_length}$.

Product Squared The prior is proportional to $(\frac{1}{component\_length})^2$.

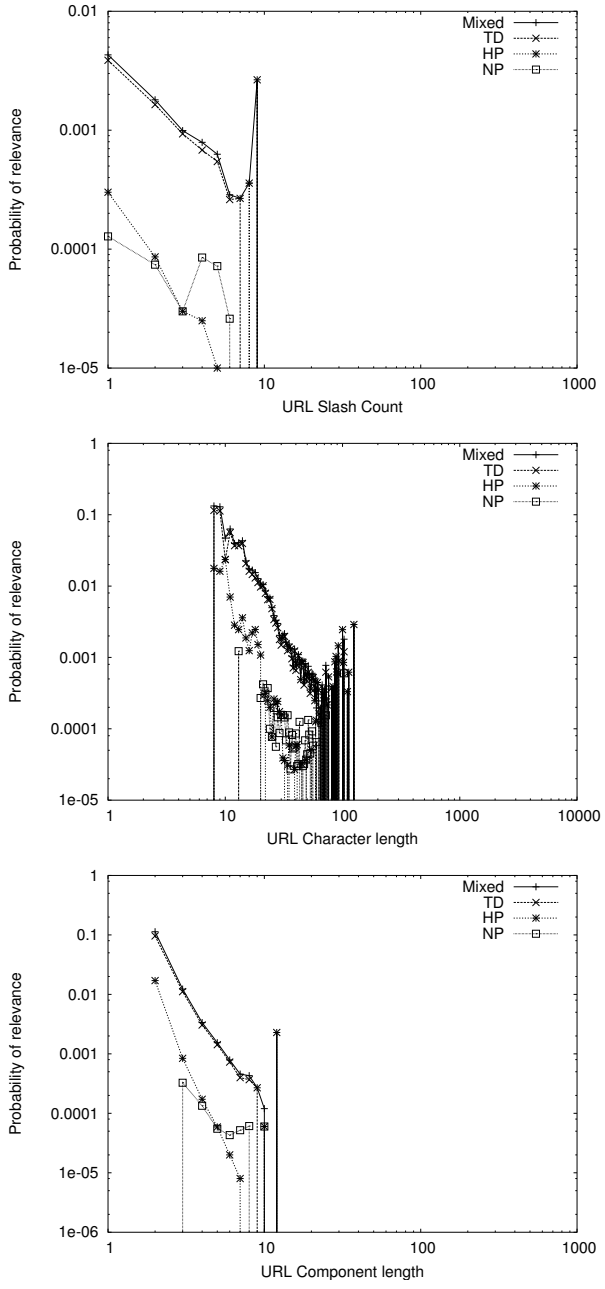

Figure 2: URL length in terms of slashes (top), characters (middle), and 'components' (bottom).

On pre-submission experiments using TREC 2003 data, the product squared prior proved to be the most effective, so we decided to use it for our official 2004 submissions.

2.2.3 Link Topology

Now, we will focus on the link topology. We restrict our attention to the indegree and outdegree of pages:

Indegree the number of pages linking to a document, and

Outdegree the number of pages to which a document links.

Figure 3 shows the prior probability of relevance over indegree and outdegree. The degree of a page has a clear relation with relevancy: the more links a pages receives, or the more pages it links to, the more likely it is that the page is relevant. Pages with many inlinks are generally good authorities, and pages with many outlinks are generally good hubs.

We used three operationalizations of the priors.

Indegree The prior is proportional to the indegree.

Log Indegree The prior is proportional to the log of the indegree.

Outdegree The prior is proportional to the outdegree.

Log Outdegree The prior is proportional to the log of the outdegree.

Pre-submission experiments on the TREC 2003 data set gave the best results for the plain Indegree prior. So we decided to use the Indegree prior in our official 2004 submissions.

2.2.4 Implementing the Priors

For the implementation of the prior probability of the documents, we face a choice of method:

Within the Language Model An elegant way to implement the prior is directly in the language modeling scoring formula (see §2.1). This implies that the corresponing prior for all documents in the collection needs to be calculated, and is being fed into the language model. The result set consists of the 1,000 documents with the highest final score, based on both the content and the prior.

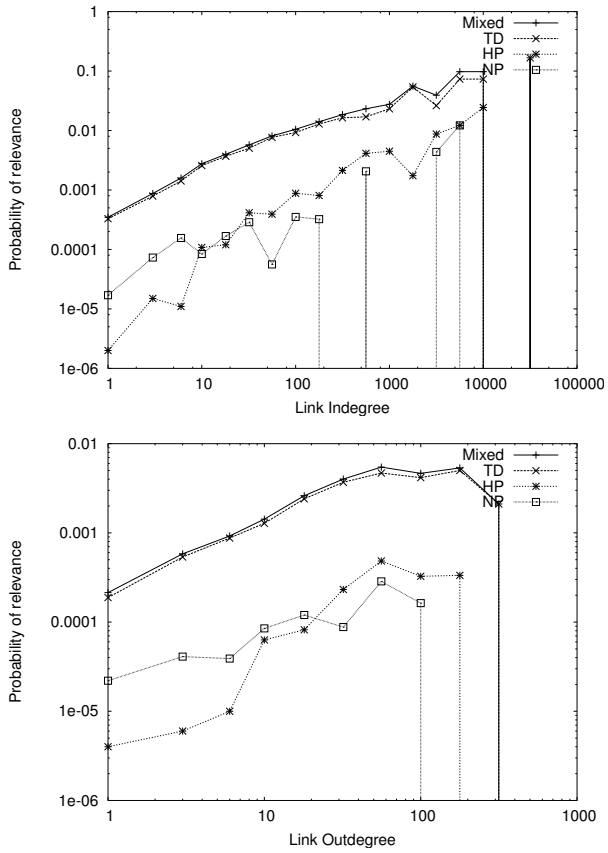

Figure 3: Link indegree (top) and link outdegree (bottom).

Reranking Prior Alternatively, one may argue that the prior should not influence what pages are returned, but only influence the relative ranking of pages returned because of their content. This can be realized in the following way: a content-based run is produced not using the prior, and the score is recalculated by multiplying the content-based score with the prior probability. The result set now consists of the 1,000 documents with the highest content-based score, reranked according to the final score.

For some priors, the reranking implementations is much more effective. Consider, for example, the case of an indegree prior. Here, the indegree can be fairly large number (ranging from 1 to 44,499), causing the infiltration of pages with a very low content-score, but a very high indegree. For our official runs, we used the priors as a reranking of an original, content-based result set.

2.3 Runs

We created two "base" runs using the mixture language model (see §2.1) on either the three stemmed indexes, or the three non-stemmed indexes:

`UAmsT04MW` Mixture Language models on the non-stemmed indexes (Full-text, Anchors, Titles), $\lambda = 0.3$

`UAmsT04MS` Mixture Language models on the Snowball [12] stemmed indexes (Full-text, Anchors, Titles), $\lambda = 0.1$

The word-based run is geared toward precision, hence the higher value of the smoothing parameter.

These two base runs were reranked with either an

Indegree prior the prior probability of a document is proportional to *indegree*, or an

URL-length prior the prior probability of a document is proportional to $(\frac{1}{component\_length})^2$.

This resulted in the following four runs:

`UAmsT04MWind` Mixture language models on the non-stemmed indexes (Full-text, Anchors, Titles), $\lambda = 0.3$, using an indegree prior.

`UAmsT04MWurl` Mixture Language models on the non-stemmed indexes (Full-text, Anchors, Titles), $\lambda = 0.3$, using an URL prior.

`UAmsT04MSind` Mixture language models on the stemmed indexes (Full-text, Anchors, Titles), $\lambda = 0.1$, using an indegree prior.

`UAmsT04MSurl` Mixture Language models on the stemmed indexes (Full-text, Anchors, Titles), $\lambda = 0.1$, using a URL prior.

The run labeled `UAmsT04MSind` was one of our official 2004 submissions.

The same URL-length prior has been applied to the indegree prior runs:

UAmsT04MWinu Mixture Language models on the non-stemmed indexes (Full-text, Anchors, Titles), $\lambda = 0.3$, using an indegree prior, and an URL prior.

UAmsT04MSinu Mixture Language models on the stemmed indexes (Full-text, Anchors, Titles), $\lambda = 0.1$, using an indegree prior, and an URL prior.

The runs labeled UAmsT04MWinu and UAmsT04MSinu were both part of our official 2004 submissions.

These two resulting runs were combined using CombMNZ on the non-normalized scores [2]:

UAmsT04MWScb CombMNZ (non-normalized, non-weighted) of runs UAmsT04MWinu and UAmsT04MSinu.

We also submitted the run labelled UAmsT04MWScb as an official run for 2004.

There is one further run experimenting with methods for boosting early precision in the vector space model:

UAmsT04LnuNG We make use of phrase and proximity operators applied to word n-grams from the topic, and use different representations of the document which are likely to contain phrases such as keywords, title, propagated anchor text, etc. The reranking methods for exploiting indegree and URL length are the same as our language modeling runs.

The run labeled UAmsT04LnuNG completes our set of official submissions for 2004.

2.4 Results

Before we discuss our results for the mixed query task, we present the results for a breakdown of the set of topics into the three subtasks, i.e., topic distillation, home page finding, and named page finding.

2.4.1 Topic Distillation

The results for the topic distillation subtask are shown in Table 3 (best scores in boldface). The second column gives the mean average precision score, the three remaining columns the percentage of topics with at least one relevant document in the top 1, top 5, or top 10. For topic distillation, we make the following observations. First, all priors (URL, indegree, and combined prior) pay off, leading to impressive improvements over the content-based scores. In particular, the indegree prior makes a substantial difference. Second, the differences between the stemmed and non-stemmed indexes are not very large, with the stemmed indexes slightly superior for most of the scores. Finally, the run using query word n-grams tailoring for precision received, with distance, the best score for success at 1.

Table 3: Results for topic distillation.

Run identifier	MAP	S@1	S@5	S@10
UAmsT04MW	0.0980	0.1733	0.3867	0.5600
UAmsT04MS	0.0973	0.1733	0.4133	0.5333
UAmsT04MWurl	0.1118	0.1867	0.4133	0.6133
UAmsT04MSurl	0.1169	0.1867	0.4667	0.6400
UAmsT04MWind	0.1310	0.3067	0.6400	0.7333
UAmsT04MSind	0.1328	0.2933	0.6533	0.7600
UAmsT04MWinu	0.1418	0.3467	0.6533	0.7733
UAmsT04MSinu	**0.1462**	0.3733	**0.7200**	**0.7867**
UAmsT04MWScb	**0.1462**	0.3600	0.6667	0.7600
UAmsT04LnuNG	0.1447	**0.4267**	0.6667	0.7467

2.4.2 Home Page Finding

The results for the home page finding subtask are shown in Table 4 (best scores in boldface). For this task, we find the following. Firstly, as with the earlier topic distillation task, for this task the priors pay off as well. There is a substantial improvement for both the URL and indegree prior. The best MAP score is for the combined prior, although the result is very close to the result of the indegree prior only. Secondly, the runs on the stemmed indexes are

Table 4: Results for home page finding.

Run identifier	MAP	S@1	S@5	S@10
UAmsT04MW	0.4245	0.2933	0.6133	0.7200
UAmsT04MS	0.4426	0.3200	0.6000	0.7200
UAmsT04MWurl	0.5719	0.4667	0.6933	0.7867
UAmsT04MSurl	0.5863	0.4800	0.7067	0.7600
UAmsT04MWind	0.6411	0.5467	0.7333	0.7867
UAmsT04MSind	0.6508	**0.5600**	0.7467	0.8267
UAmsT04MWinu	0.6374	0.5200	0.7733	0.8267
UAmsT04MSinu	**0.6553**	**0.5600**	0.7600	0.8267
UAmsT04MWScb	0.6430	0.5200	**0.7867**	**0.8400**
UAmsT04LnuNG	0.5745	0.5333	0.6400	0.6800

generally somewhat better than those on the non-stemmed indexes. Finally, the scores obtained here are, in an absolute sense, much higher than for the distillation topics. This implies that the home page topics will have a larger impact on the MAP score over all mixed queries.

2.4.3 Named Page Finding

The results for the named page finding subtask are shown in Table 5 (best scores in boldface). For the named page finding task, we see the following. First, the performance of the plain mixture model runs (with a uniform prior) is impressive with over 80 percent of the topics in the top 5. The performance is much higher than the plain mixture model runs for the other known-item search task, home page finding. Second, the priors are much less effective than for the distillation and home page finding topics. The results for the priors are mixed at best: the URL prior leads still to a slight gain in performance, but indegree and combined prior lead to a loss of performance. Thirdly, although the differences are small, the runs on the non-stemmed indexes are generally somewhat superior to the stemmed indexes. Finally, also the scores for the second known-item task are, in an absolute sense, much higher than for the distillation topics. This implies that the home page finding and named page finding topics will dominate the MAP score over all mixed queries.

Table 5: Results for named page finding.

Run identifier	MAP	S@1	S@5	S@10
UAmsT04MW	0.6567	0.5733	0.8000	0.8667
UAmsT04MS	0.6512	0.5467	**0.8133**	0.8667
UAmsT04MWurl	0.6629	0.5733	**0.8133**	0.8667
UAmsT04MSurl	**0.6769**	**0.6000**	0.7867	0.8533
UAmsT04MWind	0.6308	0.4933	**0.8133**	**0.8800**
UAmsT04MSind	0.6274	0.5067	0.8000	0.8667
UAmsT04MWinu	0.5974	0.4533	0.8000	0.8667
UAmsT04MSinu	0.5923	0.4533	0.7600	0.8400
UAmsT04MWScb	0.6098	0.4667	**0.8133**	0.8667
UAmsT04LnuNG	0.4166	0.3067	0.5867	0.6533

2.4.4 Mixed Query Task

We now discuss the results of the whole set of mixed query topics. The results are shown in Table 6 (best scores in boldface). For the entire set of mixed query topics, we see the following. First of all, the priors help to improve retrieval effectiveness. The indegree only prior is the most effective and gets the highest MAP score. The combined priors get a slightly lower MAP score, but slightly higher success at 1, 5, and 10 scores. Second, the stemmed indexes are slightly superior to the non-stemmed indexes, although the differences are small. Finally, the overall performance of the retrieval system is impressive with an MAP of close to 0.5, and over 80% of the topics with at least one relevant page in the top 10.

Table 6: Results for mixed queries.

Run identifier	MAP	S@1	S@5	S@10
UAmsT04MW	0.3930	0.3467	0.6000	0.7156
UAmsT04MS	0.3970	0.3467	0.6089	0.7067
UAmsT04MWurl	0.4489	0.4089	0.6400	0.7556
UAmsT04MSurl	0.4600	0.4222	0.6533	0.7511
UAmsT04MWind	0.4677	0.4489	0.7289	0.8000
UAmsT04MSind	**0.4703**	0.4533	0.7333	0.8178
UAmsT04MWinu	0.4589	0.4400	0.7422	**0.8222**
UAmsT04MSinu	0.4646	**0.4622**	0.7467	0.8178
UAmsT04MWScb	0.4663	0.4489	**0.7556**	**0.8222**
UAmsT04LnuNG	0.3786	0.4222	0.6311	0.6933

2.4.5 Conclusions

Two web-centric techniques, the use of URL structure and the use of web topology, were shown to be effective for the mixed query task. The break down of the task in topic distillation, home page finding and named page finding, revealed that these techniques are particularly helpful for distillation and home page topics, but give mixed results for the named page topics. In terms of mean average precision, topic distillation is a much harder task than the known-item searches. This implies that the MAP for the known-item topics will also dominate the mixed queries score, and that a system tuned for known-item search may easily outcompete a generic web retrieval system. For the success at *n* measures, all topic types contribute equally; hence, for the mixed queries the success at *n* scores seem to be the best performance indicators for this task.

3 Terabyte Track

We performed some initial experiments for the Terabyte track, aiming to test the scalability of some of the tech-

niques proven effective for the smaller web collections.

3.1 Indexes

For the .GOV2 collection, we built the following two indexes:

Titles Snowball stemmed index of all ⟨title⟩ fields. The index contains all 25,205,179 documents, although only 20,919,902 have text (after removing stopwords). Thus, the index covers 83% of the total collection.

The indexing proper took 240 minutes, preprocessing took ± 5 days to extract the titles from the collection. The total size of the index is 1,406 MB. An exhaustive run takes 17 minutes and 21 seconds for all 50 title-only topics.

Anchors Snowball stemmed index of all incoming anchor-texts, only considering fully specified URLs, i.e., http://xxx.yyy/zzz. We only index the anchor-text (if present, some links are on non-text), and ignore the ALT fields. We only index a single occurrence of repeated anchor-texts.

These are all between-site links plus only verbose within-site links; most within-site links are ignored. Contains in total 1,643,078 documents, although only 1,507,499 have text (after stopping). Thus, this covers in total 6% of the total collection.

The indexing proper took 23 minutes, preprocessing took ± 5 days for anchor-text extraction, and ± 10 hours on generating the propagated anchor-text documents. The total size of the index is 105.6 MB. An exhaustive run takes 33 seconds for the 50 title-only topics.

Based on the extracted anchor-texts (non-sorted), we calculated the within-collection indegree. This indegree can be used as a prior in the following way. As with the Web Track experiments, we use a prior that is proportional to the indegree. However, since the indegree can be fairly large number (ranging from 1 to 1,834,555), this may cause the infiltration of pages with a very low content-score, but a very high indegree. Thus, we decided to apply the prior as a reranking post-processor (see §2.2.4). Since reranking the top 10,000 documents will effectively allow the infiltration of almost any page with a very low content-score, we decide to only "rerank" the top 100 documents. Since we calculate the actual probabilities in the mixture model (as detailed in §2.1), we can simply multiply by the degree (without dividing with the sum of all degrees). Since we now multiply with a number that is larger or equal than one, we will never get a lower similarity score by applying the prior. Now, we'll only apply the length prior to the 100 documents with the highest content-based similarity score. At ranks 101 through 10,000, the documents remain ranked according to the content-score only.

3.2 Runs

We submitted the following five runs, all using only the title field of the topics:

UAmsT04TBtit Language model run on the stemmed titles with $\lambda = 0.7$ and length-prior.

UAmsT04TBanc Language model run on the stemmed anchors with $\lambda = 0.7$, and length-prior.

UAmsT04TBm1 We use a mixture language model (see §2.1) run on the stemmed titles and anchors, with $\lambda = 0.1$ and no length-prior. We use the titles index as the collection model.

UAmsT04TBm3 Mixture language model run on the stemmed titles and anchors, with $\lambda = 0.3$ and no length prior.

UAmsT04TBm1p Mixture language model run on the stemmed titles and anchors, with $\lambda = 0.1$ and no length prior, using an indegree prior on the top 100 documents per topic.

3.3 Results

At the time of writing, the results for the Terabyte track are not yet available.

4 Question Answering Track

Following the modification of the QA task this year, we give separate accounts of our approaches for answering factoid/list questions and for answering "other" questions.

First, though, we address a complication in the presentation of questions in this year's QA task: the grouping of questions by *target*.

4.1 Handling targets

Each target is given explicitly as a phrase, and the questions for the target are presented in sequence. The possibility of anaphoric dependencies of the questions on the target or on preceding questions is thus introduced. We use an anaphoric resolution module to resolve pronouns occurring in the questions. Our module is simple: each pronoun is resolved to the highest ranked compatible antecedent in the antecedent list. The antecedent list consists of the target and all noun chunks occurring in preceding questions (with previously resolved pronouns replaced by their antecedents).

Our heuristic is to rank the target highest and to rank the other noun chunks according to their occurrence order. Compatibility is determined according to a simple type system: pronouns are marked **human** (*he*, *she*, etc.), **non-human** (*it*, *this*, *that*, etc.), or **unknown** (*they*, etc.), while other noun chunks are unmarked until they are resolved to a pronoun. These simple heuristics appear to work well with the question groups, where few entities are introduced and where there is a strong tendency to refer to the target.

4.2 Factoid Questions

Our approach for answering factoid questions is largely based on QUARTZ, our QA system used for experiments in the TREC 2003 QA track [7] and the CLEF 2004 Question Answering track [6]. We use an architecture where several streams run in parallel: each is based on a different approach to QA and is a self contained QA system in itself. A final step of merging the results of the streams is based on both redundancy of answers between streams and a process of learning the strengths and weaknesses of each of them [3].

This year, apart from minor technical modifications of these streams, we employed two new components in QUARTZ: an additional stream exploiting an open-domain encyclopedia and a mechanism for type checking of the answer candidates generated by each stream; see Figure 4. We also implemented a number of simple filtering mechanisms that serve as sanity checks for the answer candidates and performed a number of experiments in our answer justification module. We give an overview of the new components here and refer the reader to [3, 4, 6, 7] for an account of the rest of the system.

Encyclopedia Stream

Many systems participating in the TREC QA track use not only the local (AQUAINT) corpus, but also additional knowledge sources such as the web and various gazetteers [15]. The use of external resources (such as the web) in QUARTZ has proved to be beneficial, and we have therefore decided to employ an additional source of external knowledge into the system: a corpus specifically designed to address the information needs expressed by the open-domain questions appearing in TREC—an encyclopedia.

We used the English edition of Wikipedia (`http://en.wikipedia.org`), a free-content encyclopedia: among the reasons to use it are its relatively wide coverage, its availability in a standard database format, and the fairly structured format of its entries.

We adopt a simplification of techniques reported already in [10]. Given a question and the question topic, we first extract the Wikipedia entry for the topic. The QUARTZ question classifier module identifies the named entity type that should be returned as an answer to the question; a named entity tagger then identifies potential answers in the encyclopedic entry. The list of answers is then ranked according to two factors: the *prior answer confidence*, which is an estimate of how likely it is that the named entity is an answer to any question, and a *posterior answer confidence*, which is an estimate of the likelihood of the named entity to be an answer to the question at hand. For estimating the prior confidence, we use layout information about the Wikipedia format, basically giving more confidence to named entities appearing earlier in the entry; for the posterior estimations, we calculate a sentence-level similarity score between the question and sentence containing the answer, based on the Jaccard measure. The final ranking of the answers is a combination of the prior and posterior estimations, with more weight given to the posterior one.

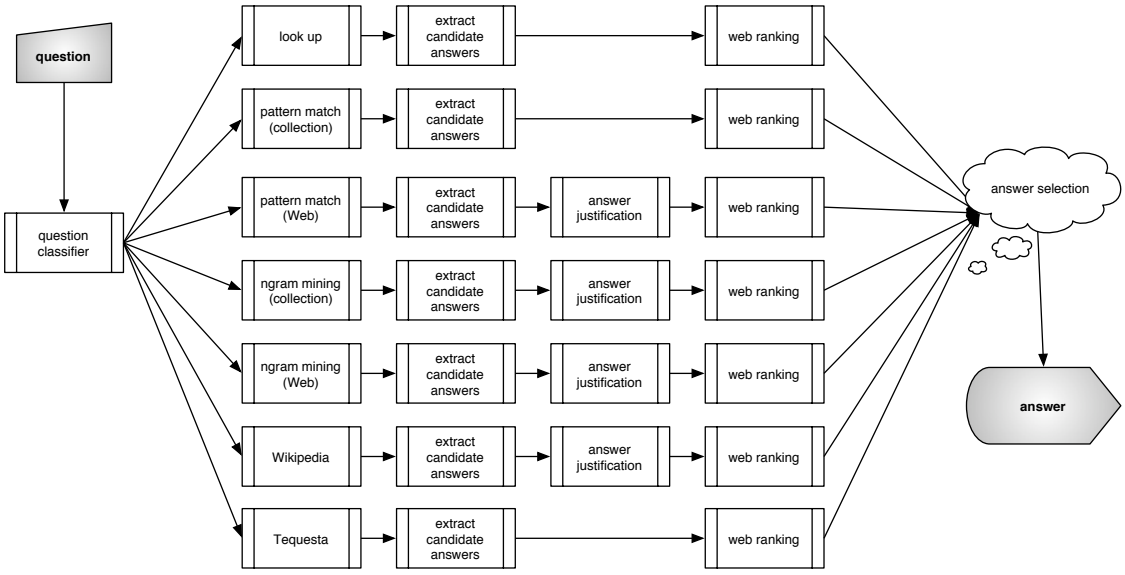

Figure 4: QUARTZ System Overview.

Answer Type Checking

In question analysis, search, and extraction of answer candidates, QUARTZ, like other QA systems, applies a recall oriented strategy. The underlying assumption is that recall can be maintained at an acceptable level in the early steps of the QA process because possible noise will be filtered out in the final filtering step.

Answer type checking—checking whether a given answer candidate belongs to the expected semantic type (or set of types)—is one filtering method that we explored further in this year's TREC evaluation. The factoid questions used in the TREC QA track are associated with a small number of semantic types—the expected types of the correct answers. On top of the coarse-grained expected answer types used to extract answer candidates (such as PERSON, LOCATION, DATE or ORGANIZATION), we found it useful to identify more precisely whether we are looking for, e.g., an ACTOR, a CAPITAL, a YEAR, or an NGO (Non-Governmental Organization).

We extended previous experiments in domain-specific type checking [11] to an open-domain type checker by combining two fitering approaches, *ontology-based* and *redundancy-based*. First, we extract a WordNet synset as the required expected answer type of a question. For each candidate answer, we then calculate the probability that it has an expected type, based on word co-occurrence of the answer and the expected type on the web. Finally, an answer is filtered unless it is more likely to be of the expected answer type than of one of its WordNet siblings. Due to space restrictions, a more detailed description and a formal evaluation of this new type checker will be published elsewhere.

Additional Filtering

In previous evaluations, we have encountered the problem of "junk"—ungrammatical answers resulting mainly from the combination of different streams and the heavy usage of n-gram techniques in QUARTZ [5]. This year we employ a simple *web hit count filter* to cope with this phenomenon. Each candidate answer is sent, as a phrase search, to Google, and phrases that have no results at all are considered to be incorrectly formed answers and removed from the candidate list.

Additionally, since our system also relies on external knowledge such as the web and Wikipedia, it often obtains answers even for questions with no answer in the

collection (NIL questions). We attemp to detect such questions using another simple filter, a *collection hit count filter*: each question topic is searched for in the collection and, if no documents are retrieved, a NIL response is given for the question.

Answer Justification

An additional substantial problem found in previous evaluations is that of *unsupported answers*, i.e., correct answers with an incorrect supporting document. We have invested some effort in improving our answer justification mechanism, with an improvement of more than 20% on training data; even so, unsupported answers still account for half of our total number of correct answers.

Previously, we used Okapi-based retrieval for answer projection, with the query formed from the question and the answer. The Okapi model's good performance on early precision allowed us to take the top retrieved document as the supporting document. For this year, we still base our projection mechanism on retrieval only, but have moved from Okapi to a vector space model with extensive usage of various query operators in the query. We issue the answer as a phrase term, identify phrases in the question and issue them as phrase terms as well, and use boolean operators for various terms in the query. These are techniques that are known to increase early precision, and, as mentioned, we have indeed noticed an improvement on the training data.

4.3 List Questions

As in the previous TREC QA track, we have not implemented a specific mechanism to handle list questions, but rather used our factoid approach for these questions, as well. The top ranking answers according to this approach are given as the answer to the list question; the number of answers depends on a confidence drop in the scores assigned to the candidates; in the absence of such a drop, a fixed threshold is used.

4.4 "Other" Questions

The QA track at TREC 2003 [15] presented an interesting new challenge: answering definition questions. Our approach to this used feature-based location and mining of web pages containing prominent information about the entities in question [7, 14]. This year with the introduction of "other" questions—similar to definition questions—we shifted our focus from web-based summarization to corpus-based generation of answers. In a nutshell, given a topic, we first obtain a set of "important facts" using a reliable, external source and then use this set to rank text segments from the AQUAINT corpus that contain the topic. In this section we provide a more detailed account of this approach.

We split the task of providing key facts about an entity from a large corpus into two stages. The first stage involves locating facts regarding the entity in the corpus. The second includes sorting these facts by order of importance, distinguishing between facts that are key facts and facts that are relatively unimportant. We approach these tasks as follows.

Extracting Facts from the Local Corpus

To obtain a list of facts about the entity, we first retrieve all documents in the collection containing the entity (as a phrase). Each document is indexed with two separate representations: headline and body text. The retrieval score is a combination of the retrieval using the two representations, assigning a higher weight to the headline representation. Next, for each document of the retrieved documents, we extract facts regarding the entity. We resolve pronouns in the document using the same anaphora resolution module applied to the questions (see §4.1), but instead of resolving each pronoun to the single highest ranked compatible antecedent, we resolve it to a disjunction of all compatible antecedents. (We do not expect our simple ordering heuristic to work as well for the more complicated discourses found in the corpus as it does for the question groups, so rather than experiment with the wide variety of suggested heuristics (see, e.g., [13]), we sidestep the issue entirely.) Then, we extract all sentences which contain the entity (either originally or after the resolution). Finally, we segment the sentences into *fact nuggets*: each sentence is parsed and converted to a list of predicate-argument snippets (essentially, each snippet contains a verb with all its arguments and modifiers)

All extracted facts are given a *prior importance estimation*, based on the retrieval score of the document containing them.

Extracting Facts from an External Source

For this stage, we turn to an external knowledge source that is likely to contain important facts about entities. A natural candidate for such a source is an open domain encyclopedia; we again make use of the English edition of Wikipedia mentioned earlier. We also considered (but did not use in the reported experiments) other highly reliable information sources, e.g., biography pages from `biography.com`.

Given a topic, we extract the encyclopedia entry for this topic, and repeat the process described in the previous section for extracting facts (i.e., anaphora resolution and sentence splitting). We are then left with a list of facts which are important enough to be included in an entry in an encyclopedia (we leave out technical details of cleaning up the data, for example removing "user added comments" common in the Wikipedia). Every fact is assigned a *posterior importance estimation* according to some layout cues, such as its proximity to the beginning of the encyclopedia entry, its placement in a table, etc; this is based on examining the Wikipedia entries and noting that like most encyclopedia entries, important information is listed first, tables usually contain key facts, and so on.

Ranking Facts by Importance

Finally, we have a list of ranked facts from our corpus (*local facts*), and an additional list of ranked important facts from an external corpus (*external facts*). To rank the local facts by means of the external ones, we measure the sentence-level similarity between local and external facts using both information-theoretic measures such as Jaccard and word-overlap, and linguistically-motivated measures such as those discussed in [1]. Using the prior and posterior estimations described earlier, we derive the final importance estimation of a fact as the product of the estimations and the sentence similarity. We then sort the facts in decreasing order of importance and provide the top *N* as the final response of the system.

Rewriting "Other" Questions as Factoids

In addition to the text nugget reranking mechanism described, we employ an additional approach for answering the "other" questions that was used by some systems (including QUARTZ) in TREC 2003 to answer definition questions. Given a topic, we generate a short list of factoid questions which yield important information for this topic, submit them to our standard factoid engine, and formulate an answer nugget from the result. The templates for generating a question, as well as the templates for constructing an answer nugget, are very limited (3-4 per topic) and hand-crafted; to select the right set of templates, we distinguish between a number of entity types: PERSON, LOCATION, ORGANIZATION, EVENT, ARTIFACT. A sample template/answer pair for the PERSON category is `When was TARGET born? TARGET was born on ANSWER`. The nuggets generated by this method are combined with the previous, reranked nuggets with a simple duplication removal mechanism. To classify a topic as an entity type, we use a number of heuristics including WordNet lookup, resolution of a **human** pronoun (such as *he* or *she*) to the topic (as evidence for the PERSON category), and an NE recognition mechanism used also elsewhere in QUARTZ.

4.5 Runs

We submitted 3 runs differing only in the final sanity checking for answer candidates. Our aim here was to compare different options for the answer filtering.

`uams04raw` No answer type checking or other filtering mechanisms employed.

`uams04tc1` Answer type checking and web and collection hit-count filters described above used.

`uams04tc2` Same as previous run, but Wikipedia not used for the 'other' questions.

4.6 Results

Table 7 gives the combined results for the 3 QA tasks (accuracy for factoids, F score for list and definition questions) and the final scores of our runs.

The results are disappointing, especially in light of our recent good performance for Dutch Question Answering [6] (where the questions are easier than the TREC questions, and rather similar to the TREC8 or TREC9 QA track). A preliminary error analysis shows that most errors are due to the insufficiently fine-grained question

Table 7: Results for the QA track

Run identifier	A (Exact,Lenient)	F (List)	F (Def)	Overall
uams04raw	0.135 , 0.287	0.094	0.210	0.143
uams04tc1	0.126 , 0.269	0.085	0.207	0.136
uams04tc2	0.126 , 0.269	0.087	0.184	0.131

classification and entity extraction (the current system uses only 37 question types and 5 NE types). Also, we note the high rate of unsupported answers: without the answer justification requirement, our performance would double. Furthermore, our type-checking module does not seem robust enough to improve the performance of the system.

4.7 Conclusions

In general, we found that the new setting of the task (groups of related questions, often with anaphoric references) represents a tractable (and appealing!) problem. The biggest sources of errors for our system are still in the "core" QA part: detecting expected answer type and extracting candidates. New challenges, e.g., inter-question anaphora resolution, have been addressed fairly successfully, and other new aspects, e.g., the presence of explicit topics, even seem to help locate relevant information. The use of external resources (Web, Wikipedia) does improve the performance of the system but poses additional problems for projecting externally found answers into the collection. In our future work we plan to refine the answer type identification and entity extraction modules, and place an emphasis on highly reliable semi-structured encyclopedia-like resources available on the Web.

5 Conclusions

In this paper we have described our participation in the TREC 2004 Web, Terabyte, and Question Answering tracks.

For the Web track, our findings highlighted that web retrieval is unlike standard ad hoc retrieval. Whereas document-length is a useful indicator for relevance in the general ad hoc case, it is not for the case of web retrieval. Specific webcentric techniques, such as using the URL structure or using the link topology, turned out to be useful indicators of relevance for the mixed query task. These web-centric techniques were particularly useful for topic distillation and home page finding, but not so for named page finding. This can be easily explained by the task definition that required returning home pages of sites for both topic distillation and named page finding.

For the Terabyte track, it is too early to formulate conclusions, as evaluation results are not yet available at the time of writing.

This year, our work for the Question Answering track was largely motivated by the wish to extend our QA system to handle the new, more complex question presentation, and to see how our existing modules cope with this new setting. While the new setting proved tractable, a variety of bugs in the "core" of our QA engine lead to rather disappointing scores.

Acknowledgments

Thank you to Börkur Sigurbjörnsson for useful suggestions and discussion.

This research was supported by the Netherlands Organization for Scientific Research (NWO) under project numbers 017.001.190, 220-80-001, 264-70-050, 612-13-001, 612.000.106, 612.000.207, 612.066.302, and 612.069.006.

References

[1] M. D. Boni and S. Manandhar. The use of Sentence Similarity as a Semantic Relevance Metric for Question Answering. In *Proceedings of the AAAI Symposium on New Directions in Question Answering*, 2003.

[2] E. Fox and J. Shaw. Combination of multiple searches. In D. Harman, editor, *The Second Text REtrieval Conference (TREC-2)*, pages 243–252. National Institute for Standards and Technology. NIST Special Publication 500-215, 1994.

[3] V. Jijkoun and M. de Rijke. Answer selection in a multi-stream open domain question answering system. In S. McDonald and J. Tait, editors, *Proceedings 26th European Conference on Information Re-*

trieval (ECIR'04),, volume 2997 of *LNCS*, pages 99–111. Springer, 2004.

[4] V. Jijkoun, M. de Rijke, and J. Mur. Information extraction for question answering: Improving recall through syntactic patterns. In *Proceedings of the 20th International on Computational Linguistics (COLING 2004)*, 2004.

[5] V. Jijkoun, G. Mishne, and M. de Rijke. How frogs built the Berlin Wall. In *Proceedings CLEF2003*, volume LNCS. Springer, 2004.

[6] V. Jijkoun, G. Mishne, M. de Rijke, S. Schlobach, D. Ahn, and K. Müller. The university of amsterdam at qa@clef 2004. In C. Peters and F. Borri, editors, *Working Notes for the CLEF 2004 Workshop*, pages 321–324, 2004.

[7] V. Jijkoun, G. Mishne, C. Monz, M. de Rijke, S. Schlobach, and O. Tsur. The University of Amsterdam at the TREC 2003 Question Answering Track. In *Proceedings TREC 2003*, pages 586–593, 2004.

[8] J. Kamps, C. Monz, M. de Rijke, and B. Sigurbjörnsson. Approaches to robust and web retrieval. In E. M. Voorhees and L. P. Buckland, editors, *The Twelfth Text REtrieval Conference (TREC 2003)*, pages 594–599. National Institute of Standards and Technology. NIST Special Publication 500-255, 2004.

[9] W. Kraaij, T. Westerveld, and D. Hiemstra. The importance of prior probabilities for entry page search. In K. Järvelin, M. Beaulieu, R. Baeza-Yates, and S. H. Myaeng, editors, *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 27–34. ACM Press, New York NY, USA, 2002.

[10] J. Kupiec. Murax: a robust linguistic approach for question answering using an on-line encyclopedia. In *Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 181–190. ACM Press, 1993. ISBN 0-89791-605-0.

[11] S. Schlobach, M. Olsthoorn, and M. de Rijke. Type checking in open-domain question answering. In *Proceedings of the 16th European Conference on Artificial Intelligence (ECAI 2004)*, pages 398–402. IOS Press, 2004.

[12] Snowball. Stemming algorithms for use in information retrieval, 2004. `http://www.snowball.tartarus.org/`.

[13] J. R. Tetreault. A corpus-based evaluation of centering and pronoun resolution. *Computational Linguistics*, 27:507–520, 2001.

[14] O. Tsur, M. de Rijke, and K. Sima'an. Biographer: Biography questions as a restricted domain question answering task. In *Proceedings ACL 2004 Workshop on Question Answering in Restricted Domains*, 2004.

[15] E. Voorhees. Overview of the trec 2003 question answering track. In *The Twelfth Text REtrieval Conference (TREC-03)*, 2004.