JHU/APL at TREC 2001: Experiments in Filtering and in Arabic, Video, and Web Retrieval

James Mayfield, Paul McNamee, Cash Costello, Christine Piatko, and Amit Banerjee
Research and Technology Development Center
Johns Hopkins University Applied Physics Laboratory
11100 Johns Hopkins Road, Laurel, Maryland 20723-6099 USA
{mayfield, mcnamee, costello, piatko, banerjee}@jhuapl.edu

Overview

The outsider might wonder whether, in its tenth year, the Text Retrieval Conference would be a moribund workshop encouraging little innovation and undertaking few new challenges, or whether fresh research problems would continue to be addressed. We feel strongly that it is the later that is true; our group at the Johns Hopkins University Applied Physics Laboratory (JHU/APL) participated in four tracks at this year's conference, three of which presented us with new and interesting problems. For the first time we participated in the filtering track, and we submitted official results for both the batch and routing subtasks. This year, a first attempt was made to hold a content-based video retrieval track at TREC, and we developed a new suite of tools for image analysis and multimedia retrieval. Finally, though not a stranger to cross-language text retrieval, we made a first attempt at Arabic language retrieval while emphasizing a language-neutral approach that has worked well in other languages. Thus, our team found several challenges to face this year, and this paper mainly reports our initial findings.

We also made a last-minute (really a last 36 hour) effort to participate in the web retrieval track. We unearthed a year-old index and the software that we used for the web task at TREC-9, and very quickly produced some official submissions. Our main interest in the home-page finding task was to submit content-only runs that could serve as a simple baseline to which other group's sophisticated hyperlink-influenced approaches might be compared. We simply did not have the time to seriously investigate the more complex problems being examined by the web track; however, we wanted to be good TREC citizens and contribute to the document pools.

All of our text-based investigations were based on the Hopkins Automated Information Retriever for Combing Unstructured Text, or HAIRCUT system. HAIRCUT is a Java-based tool developed internally at JHU/APL that was first used to compare tokenization methods during TREC-6. HAIRCUT benefits from a basic design decision to support flexibility throughout the system. For example, the software supports words, stemmed words, character n-grams, and multiword phrases as indexing terms. And, several methods for computing document similarity are supported, though we recently have relied on probabilistic methods based on statistical language modeling techniques.

In general, we have seen better performance using language models than when using cosine-based vector scoring. In our experiments we used a linguistically motivated probabilistic model. Hiemstra and de Vries describe this model and explain how it relates to both the Boolean and vector space models [4]. The model has also been cast as a rudimentary Hidden Markov Model [15]. Although the model does not explicitly incorporate inverse document frequency, it does favor documents that contain more of the rare query terms. The similarity measure can be expressed as

$$Sim(q,d) = \prod_{t=terms} \left(\alpha \cdot f(t,d) + (1-\alpha) \cdot df(t)\right)^{f(t,q)}$$

Equation 1. Similarity calculation.

where *(1-α)* is the probability that a query word is generated by a generic language model, and *α* is the probability that it is generated by a document-specific model. *df(t)* denotes the relative document frequency of term *t*.

We conducted all of our work on a set of four Sun Microsystems workstations that are shared among our department (80 physicists, chemists, engineers, and about 25 computer scientists). Two of the machines are 4-node Sun Microsystems Ultra Enterprise 450 servers with 2.5 and 4.0 GB of physical memory, respectively; the other two machines are Sun Ultra-2 workstations with 1.25 of RAM. This cluster has 200GB of dedicated, networked disk space for use in our retrieval work.

Filtering Track

We participated in both the routing and batch tasks for the filtering track. We did not use any of the hierarchy information available with the Reuters categories for either task.

Routing Task

Our goal for the routing task was to evaluate the use of a statistical language model for routing. We submitted two runs, one based on a character n-gram (*n=6*) index (*apl10frn*) and one based on a stem index (*apl10frs*) using a derivative version of the SMART stemmer. We also created an unofficial word-based run (*apl10frw*). We simulated routing, using a modified version of HAIRCUT system to score indexed test documents using training index statistics – the statistical language model described above was used for scoring. We formed queries using 60 terms per topic that were selected from the positive batch qrels documents. Term selection was accomplished using mutual information based difference statistics with respect to the August 96 training data.

We were pleased with our official results for our first participation in this task. We were excited to participate in the "routing bet" discussion and we can report that we have 28 queries (exactly 1/3 of the queries) with ≥ 0.9 precision at 1000 docs in both our official runs. The closeness of the results indicates the choice of terms is not critical.

	Avg. prec.	# bests	# ≥ median (84 topics)
apl10frn	0.121	4	70
apl10frs	0.104	4	56
apl10frw	0.113	unofficial run	

Table 1. APL Routing Results

Batch Task

Our goal for the batch task was to evaluate the effectiveness of Support Vector Machines (SVMs) on the new Reuters data set [22]. SVMs are used to create classifiers from a set of labeled training data. SVMs find a hyperplane (possibly in a transformed space) to separate positive examples from negative examples. This hyperplane is chosen to maximize the margin (or distance) to the training points. The promise of large margin classification is that it does not overfit the training data and generalizes well to test data of similar distribution. See Hearst [3] for a general discussion of SVMs.

For the batch task, we sought to explore the effects of different parameter choices on learning with this Reuters collection. We were interested in the use of tf/idf weighted vectors vs. per-topic binary vectors; the use of radial basis function (RBF) vs. linear kernels in the SVMs; score thresholds on resulting classifier scores; and training skew factors to incur less error on positive examples. This follows earlier work in text batch filtering on the original smaller Reuters collection [2] [6]. We used the SVM-light package (version 3.50, by Thorsten Joachims [19]) to create classifiers based on the training data for classification of the test data. We used a reduced feature space for both batch submissions. For all runs, we normalized document vectors to unit length.

Our post-submission results show: tf/idf training-derived features were better than topic-specific binary ones; RBF kernels were slightly better than linear kernels; aggressive score thresholding hurt our tf/idf runs, while it helped improve our binary runs; fixed skew was not as good as the per-topic skew developed by others in the track.

Batch Using Linear SVMs with Binary Vectors

For the submitted run *apl10fbsvml* we used 200 terms derived on a per-topic basis to create binary term vectors for each document (our implementation actually created a different document vector for each topic). The terms were selected from each topic's positive qrels documents, using mutual-information-like difference statistics with respect to the August 96 training sample. Given *n* positive training documents for a topic, we randomly chose *n* potentially negative examples from the full training index, and threw away any that were actually positive. We created linear SVMs, weighting positive and negative training examples equally (-j 1 flag in SVM-light). *J* is a cost or skew factor, by which training errors on positive examples outweigh errors on negative examples (see [5]).

We then used the score of the test document using the topic SVM to decide whether to return the document. In experiments reported in the literature, SVMs scores are normally thresholded above zero. However, we had observed many training errors close to zero; many negative examples were misclassifed with a small positive score. We thus experimented with setting higher score thresholds. We debated using a small epsilon to threshold the score, but decided to try to find the "best" scores per topic automatically to maximize the 2R+ -N+ measure for the training data. While the overall approach did not work all that well, thresholding did salvage something out of these particular vectors. Unofficial runs using a zero threshold did worse, for both *j=1* and *j=5* (runs *BINLIN skew1* and *BINLIN skew5* in Table 4 below).

We do not know why this approach did not succeed. We considered trying different values of *j* to weight positive and negative examples differently. Perhaps more negative training data or a greater number of terms would improve the technique. Finally, our main intuition is that binary features are probably not appropriate for this Reuters dataset.

Batch Using RBF SVMs with TFIDF Vectors

For the submitted run *apl10fbsvmr* we used a reduced term space of 2000 terms to create all the test and training document vectors, based on all the training data. The terms were selected using the top 2000 stems by document frequency in the training set. Stems were produced using a derivative of the SMART stemmer and stopwords were not removed. We created tf/idf weighted vectors for each document and each vector was normalized to unit length. Given *n* positive training documents for a topic, we randomly chose *4n* potentially negative examples from the training index, and threw away any that were actually positive. We then trained radial basis function SVMs (using the *-t 2 -g 1* flags in SVM-light), weighting positive and negative training examples equally *(-j 1* flag in SVM-light). Using thresholds higher than zero to classify the test documents, as we did with linear kernels, proved to be a big mistake. It hurt performance significantly. Set precision was good, but set recall was terrible.

We redid this run using the same RBF models with zero as the score threshold (*RBF skew1*), and are much happier with the results. We also did some runs using weighted RBF models with *j=5* (*RBF skew5*) and similarly tried linear kernels (*LIN skew1* and *LIN skew5*). These post-hoc experiments confirm that SVMs can work well for the batch task, using either radial basis functions or linear separators with tf/idf weighted vectors normalized to unit length.

We expect there are many per-topic optimizations (such as the leave-one-out cross-validation on training data Dave Lewis used to find optimal *j* weights per topic [8]) that could dramatically improve these initial findings.

Results

	T10SU	Fbeta	SetPrec	SetRecall
apl10fbsvml	0.115	0.292	0.303	0.627
apl10fbsvmr	0.081	0.154	0.380	0.054

Table 2. Official Batch Submissions.

	T10SU	Fbeta	SetPrec	SetRecall
RBF skew1	0.283	0.459	0.546	0.437
RBF skew5	0.254	0.430	0.442	0.525
LIN skew1	0.234	0.413	0.400	0.601
LIN skew5	0.157	0.341	0.318	0.689

Table 3. Unofficial (post hoc) batch runs, unified tf/idf weighted term space.

	T10SU	Fbeta	SetPrec	SetRecall
BINLIN skew1	0.030	0.132	0.113	0.835
BINLIN skew5	0.009	0.085	0.071	0.895

Table 4. Unofficial (post hoc) batch runs, per-topic binary term space.

Summary of Batch Filtering Results

Chart 1 summarizes the results of our batch filtering experiments. SVMs with RBF kernels on TFIDF vectors and no thresholding works well, and could have performed above median compared to other official batch results. Thresholding above zero hurt for RBF SVMs on TFIDF vectors (*RBF skew1* vs. *apl10fbsvmr*). However thresholding improved a poor baseline result of linear SVMs on binary vectors (*apl10fbsvml* vs. *BINLIN skew1*).

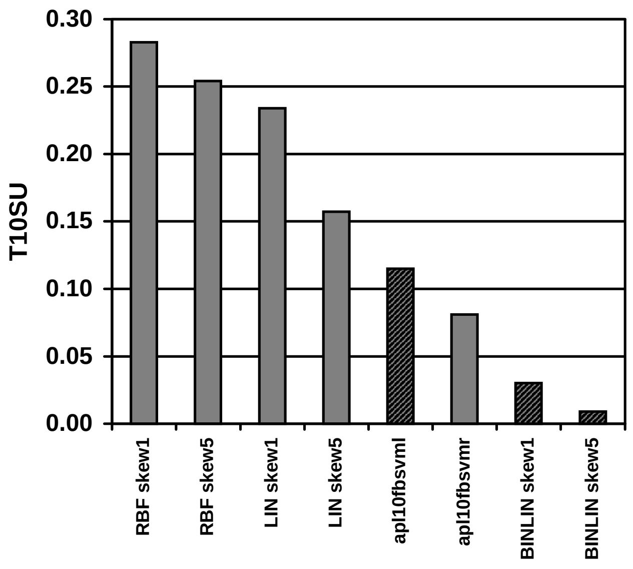

Chart 1. SVMs with RBF kernels on TFIDF vectors work well for batch filtering with the T10SU metric.

Video Retrieval

The video track was a new addition to TREC this year. It consisted of three tasks: shot boundary detection, known-item search and general search. The data set was eleven hours of mostly documentary video from the Open Video Project at University of North Carolina Chapel Hill and the NIST Digital Video Collection. JHU/APL did not have any previous experience with video or image retrieval so participation in this track was a valuable learning experience. A significant amount of time had to be devoted to developing the software infrastructure needed to process MPEG video, create an index, and parse queries. This led to a philosophy of “simple is best.”

For the shot boundary detection task, we experimented with using color histograms, luminance, and the raw image gradient of frames to locate hard cuts and gradual transitions. Hard cuts were identified using an ad hoc global threshold on the color histogram intersection of consecutive frames [16]. With gradual transitions, possible dissolves and fades were first detected by looking for abrupt changes in the average luminance of frames. These possible gradual transitions were then evaluated by analyzing the change in the image gradient. Each frame was divided into eight-by-eight blocks. If a large percentage of the blocks had changes in the image gradient greater than some threshold, the presence of a dissolve or fade was confirmed. The same technique was used to locate the start and end of each gradual transition. This approach was based on the work of Zabih, Miller, and Mai [17], the major differences being that we did not perform motion compensation and used raw image gradients rather than edges. This resulted in a method that was less computationally expensive than the typical edge entering and exiting method. The method did not perform well in the evaluation, but we did not have sufficient time to experiment with different variations and thresholds. The only interaction between the algorithm for detecting hard cuts and those for detecting gradual transitions was the hard cut algorithm taking precedence if a cut and a transition were detected in close proximity. A summary of the results for shot boundary detection is shown in Table 5.

	Total videos	# ≥ median	# best
Cuts-prec	15	12	8
Cuts-recall	15	4	1
Graduals-prec	17	0	0
Graduals-recall	17	4	2

Table 5. Shot boundary detection results

Because of limited time and experience, our approach to video retrieval was to treat a video as a series of still images. We made no attempt to exploit the extra information available with video and not with images, such as the audio track and object motion. The experiments we performed focused on using color histograms and image texture features. Each video was first decomposed into shots using the shot boundary detection algorithms described above. The middle frame was used as the key frame to represent all the content of the frames in the shot. This is not a complete representation, but it fit with the emphasis on simplicity for the sake of expediency. In fact, the index files for the 6.3 GB data set comprised only 31 MB altogether, less than 1% the size of the source data. A key frame was described by a vector that contained color and texture features. Similarly, each query was also represented by one or more of these vectors. For a description of the vectors, see Table 6.

Keyframes	Dimensions	Color features	Texture features
7391	272	256	16

Table 6. Description of video index and vector features

When processing queries, any text or audio was completely ignored. If a video example was provided for a query, just the middle frame was extracted as an image example. A weighted distance measure was used for evaluation with the key frames ranked by minimum distance to the set of query examples. The weights were chosen so that the texture and color features made approximately the same contribution to the distance measure even though there were fewer texture measures. The texture features were calculated using a texture descriptor proposed by Manjunath [9]. It creates a multiresolution decomposition using a Gabor filter bank. We used code available from the Image Processing and Vision Research Lab at the University of California, Santa Barbara [18] to calculate these features.

While our results from the known item task were close to the median, the results from the general search were significantly below average. We have not had time to completely investigate this disparity. One explanation for this is that the general information need queries depend more on the text description of the query than on the image or video examples. Since we discarded this information when parsing the query, we were at a disadvantage when trying to retrieve relevant video clips for general searches. The three queries on which we were above the median would support this hypothesis; the text descriptions were short with little information contained in them. “Other shots of city scapes,” which is the text description of a query where we were above the median, is a good example. In the known item task, the queries we scored the best on asked about objects that have a strong color component: “Scenes with a yellow boat” or “Other examples of the surface of the planet Mars.” This result agrees with the strong emphasis we placed on color in the representation of video data.

Arabic Language Retrieval

The Cross-Language Retrieval task at TREC 2001 consisted of bilingual retrieval of Arabic newspaper articles given either English or French topic statements. Monolingual submissions were also accepted using the manually produced Arabic translations of the topics.

The apparent necessity of having quality translation resources available for use in a CLIR system has often been expressed. For example, at the first CLEF workshop, Anne Diekema gave a provocative talk, suggesting that CLIR evaluation was essentially just evaluation of translation resources [1]. We spent several days searching the Web for extant collections of parallel corpora or bilingual dictionaries that would be helpful for translating to Arabic, with no real success. We finally found one newspaper that published mappable, parallel content in both Arabic and English, only to discover that the Arabic stories were available only as images (a practice that stems from the historic lack of standards and software for displaying Arabic text). Downloading that GIF files, OCRing them, and building a parallel collection was beyond our means.

Unable to discover or acquire significant translation resources, we relied exclusively on two on-line machine translation systems, Ajeeb [20] and Almisbar [21]. Recently, Kraaij showed how translation probabilities can be incorporated nicely into a language model for cross-language text retrieval, and he demonstrated the efficacy of this combination at the CLEF-2001 workshop [7]. However, since we simply used machine translation for query translation we did not have access to translation probabilities that are available when dictionaries and corpus-based approaches are used. All of our work was with fully automated retrieval.

This was JHU/APL's first experience with Arabic document processing and we learned quite a lot from the experience. We had no personnel who could read Arabic. This however, did not dampen our enthusiasm for the task in the slightest. Over the last several years, our team at APL has participated in multiple CLIR evaluations, where large document collections in Chinese, Dutch, English, French, German, Italian, Japanese, and Spanish were searched [10] [11] [12] [13] [14]. While these higher-density languages tend to have many resources available for linguistic analysis and automated translation, these languages are diverse, and use numerous character sets and character encodings. Our approach for combating the inherent scalability issues presented by working with numerous languages has been to focus on simple, language-neutral approaches to text processing. Counterintuitively, we have not found that sophisticated, linguistically-rich approaches demonstrate an appreciable performance advantage over the knowledge-light methods we espouse.

One example of a language-neutral technique is the use of overlapping character n-grams. We have found that n-grams work well in many languages and a pseudo linguistic normalization occurs in agglutinative languages such as Dutch and German [11]. N-grams are more widely used for retrieval in Asian languages; we recently showed that 3-grams perform on par with 2-grams in unsegmented Japanese text [12], which is not the case with Chinese [14]. Our use of 6-grams for indexing Arabic was not founded on linguistic principles or empirical evidence – we simply guessed that it would be a good choice as it has been in many other alphabetic languages. In retrospect, shorter n-grams have proven to work better with Arabic. In addition to examining the choice of words or n-grams as indexing terms, we experimented with eliminating or replacing certain Arabic characters that did not appear in a list of 28 letters that we had available. Thus we built four different indexes; summary information about each is shown in Table 7.

	# terms	index size
words	571,798	372 MB
words - morph	539,979	351 MB
6-grams	6,784,129	2513 MB
6-grams - morph	6,081,618	2427 MB

Table 7. Index statistics for the 869 MB, 384K article TREC-2001 Arabic collection.

Our submissions were produced by combining multiple base runs using different combinations of the topic statement fields, and different methods for morphological normalization, tokenization, query expansion, and translation. One monolingual run, three bilingual runs from English topics, and one cross-language run using the French topics were submitted. For our monolingual Arabic run, *apl10ca1*, we relied on eight constituent runs

- 2 query formats: TD and TDN
- 2 choices for relevance feedback (yes or no)
- 2 tokenization alternatives, words and 6-grams
- 1 normalization approach, character elimination was used

Thus, eight different base runs were created, and merged together to produce *apl10ca1*. See [13] for details of the merging strategy.

Apl10ce1, was our first bilingual run using the English topics. We used the exact same approach as *apl10ca1*, but had two methods for translating the topics:

- 2 translation systems (Ajeeb and Almisbar)

Thus sixteen different base runs were combined to produce the submitted run.

Our second and third English bilingual runs only made use of the TD topic fields and used either words, or 6-grams as indexing terms. The second run, *apl10ce2* used eight base runs:

- 1 query format: TD
- 2 choices for relevance feedback (yes or no)
- 1 tokenization alternative: 6-grams
- 2 normalization approachs, character elimination was used, or not
- 2 translation systems (Ajeeb and Almisbar)

The third English bilingual run, *apl10ce3*, was just like *apl10ce2*, except that words were used in place of n-grams.

Finally, we submitted one run using the French topic statements, *apl10cf1*. The base runs for this used:

- 1 query format: TDN
- 2 choices for relevance feedback (yes or no)
- 2 tokenization alternatives: words and 6-grams
- 1 normalization approach, the character elimination was used
- 2 translation systems (Ajeeb and Almisbar) from English to Arabic
- 1 translation system for French to English (Systran)

Thus, when using the French queries, we first translated to English using the Systran product, and then translated to Arabic using one of the two online systems (Ajeeb/Almisbar). Interestingly, this second layer of translation did not seem to cause much loss in retrieval effectiveness. This may be due to the generally high performance of the Systran English/French module.

Official results

An overview of APL's five official runs for the Arabic track are shown in Table 8 below.

	MAP	Recall (4122)	# best	# ≥ median	% mono
apl10ca1	0.3064	2669	3	17	100 %
apl10ce1	0.2891	2819	1	22	94.4
apl10ce2	0.2250	2593	0	16	73.4
apl10ce3	0.1914	2350	0	15	62.5
apl10cf1	0.2415	2574	0	20	78.8

Table 8. Official results for Arabic runs (25 topics)

We note that run apl10ce1 (bilingual English to Arabic) achieved 94.4% of the monolingual baseline observed in *apl10ca1*. As yet, we are unable to ascertain whether this is do in part to our particular approach to retrieval, or is more a factor of the quality of the machine translation software we relied on.

Since the conference workshop in November, we have found better bilingual performance using n-grams of length four instead of the longer six-grams. This yielded an improvement in average precision from 0.2891 (*apl10ce1*) to 0.3350. But our monolingual baseline also improved when 4-grams were used, from 0.3064 (*apl10ca1*) to 0.3588. Thus, the relative bilingual performance drops insignificantly to 93.4%.

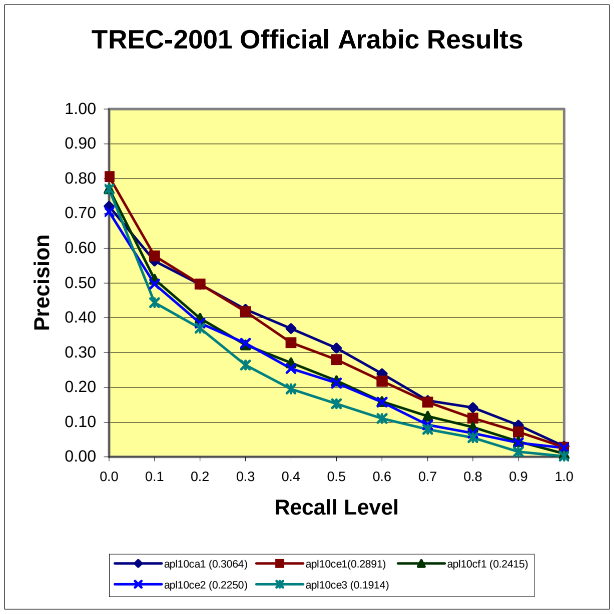

Figure 1. Recall-precision graph for APL's official Arabic track automatic submissions

We do observe that the use of n-grams accounted for a 17% relative improvement over words in mean average precision (0.2250 vs. 0.1914) as seen in the results for runs *apl10ce2* and *apl10ce3*..

We are still examining the data from all of our many base runs, and do not report on those runs. However, our preliminary analysis finds that character elimination was helpful, but the effect was not extremely large.

Web Retrieval

All of our work for this task was done in essentially one day using an index file previously created for the *wt10g* collection and used by APL during TREC-9. We submitted four content-only based runs for the ad hoc task, and produced two submissions for the homepage finding task. Our site finding runs were based entirely on query content; we did not use site popularity (backlink frequency) or any graph-theoretic analysis of the hyperlink structure. Our purpose was to see how well a pitifully under-informed approach would compare to the more sophisticated methods we anticipated others would apply to the problem.

We indexed documents using unstemmed words; the resulting dictionary contained over three million entries and the index files consumed roughly 3GB of disk space. Each document was processed in the following fashion. First, we ignored HTML tags and used them only to delimit portions of text. Thus no special treatment was given for sectional tags such as `<TITLE>` or `<H1>` and both tags and their attribute values were eliminated from the token stream. The text was lowercased, punctuation was removed, and diacritical marks were retained. Tokens containing digits were preserved; however only the first two of a sequence of digits were retained (e.g., 1920 became 19##). The result is a stream of blank-separated words. Queries were parsed in the same fashion as document, except that tried to remove stop structure from the description and narrative sections of the queries using a list of about 1000 phrases constructed from previous TREC topic statements.

After the query is parsed each term is weighted by the query term frequency and an initial retrieval is performed followed by a single round of relevance feedback. In performing blind relevance feedback we first retrieve the top 1000 documents. We use the top 20 documents for positive feedback and the bottom 75 documents for negative feedback; however duplicate or near-duplicate documents are removed from these sets. We then select 60 terms for the expanded query.

For the most part we ignored the web-nature of the documents and relied on textual content alone to rank documents.

Informational Task

We submitted four runs for this subtask, three runs that simply used the short (Title) portion of the topic statement, and one run that used all parts of the topic (TDN). The four runs were:

- *apl10wa*: title only, no blind relevance feedback
- *apl10wb*: title only, no blind relevance feedback, all query terms must be present in a document
- *apl10wc*: title only, with pseudo relevance feedback, all query terms must be present in the document
- *apl10wd*: TDN, with psuedo relevance feedback, no constraints on query term presence

	P@5	P@10	MAP	Recall (3363)	Bests / Median
apl10wa	0.1600	0.1460	.0805	1702	1 / 11
apl10wb	0.2400	0.1900	0.0671	599	1 / 9
apl10wc	0.2520	0.2380	0.1567	2105	2 / 28
apl10wd	0.3720	0.3380	0.2035	2525	2 / 30

Table 9. Performance of APL Official TREC-2001 Web submissions (Ad hoc)

Results for our official submissions are shown in Table 9. The submissions that used pseudo relevance feedback (RF) had much higher precision at 10 docs, mean average precision, and recall at 1000 docs. The run using all parts of the topic statement (*apl10wd*) had the highest performance across the board, including precision at 5 documents. Runs *apl10wb* and *apl10wc*, both of which required all query terms (only terms from the topic titles) to be present in returned documents, had about a 50 percent improvement in precision at 5 documents over *apl10wa*. This is important, because it suggests that when high precision is desirable, not all documents containing any query term need be examined, a practice common to many web search engines today (instead, the smaller set of documents that contain <u>all</u> of the query terms could be scored). Also, while *apl10wc* had high performance at higher recall levels than did *apl10wb*, this was not really true at high precision. This lends support for the practice of not using relevance feedback when only few relevant documents are needed to satisfy a user's need.

Navigational Task

We submitted just two runs for this subtask, and decided to see how well a purely content-based ranking would perform. As in the informational task, we compared performance between runs where all of the query terms were required to be present in relevant documents. We simply ordered our ranked list of hyperlinks using the similarity scores from the retrieval process. As was mentioned earlier, no use of document popularity or hyperlink structure was attempted. The two runs we submitted were:

- *apl10ha*: all terms required, no relevance feedback
- *apl10hb*: all terms not compulsory, no blind relevance feedback used

	MRR	% top 10	% failure
apl10ha	0.238	44.8%	22.1%
apl10hb	0.220	42.8%	21.4%

Table 10. Performance of APL Official TREC-2001 Web submissions (site finding task)

On the officially reported measures, mean reciprocal rank, percent of topics with a correct entry page found in the top 10 documents, and the failure percentage (when none was found in the top 100 docs), these two runs were virtually identical. The mean reciprocal rank is just slightly higher for apl10ha, in which all query terms were required to be on the given page.

Conclusions

This year we participated in three tracks that each presented new challenges: filtering, video, and Arabic.

We investigated the use of Support Vector Machines (SVMs) for batch text classification and noticed a large sensitivity to parameter settings for these classifiers. We also found that we were able to choose reasonable score thresholds for the routing task when using a language model for estimating document relevance.

Due to a lack of experience with multimedia retrieval (e.g., we had never previously participated in the TREC Spoken Document Retrieval task), the video track was a significant challenge for us. We placed an emphasis on simple techniques to quickly create a retrieval system while planning to add more advanced components such as speech recognition in the future. From our initial analysis, there was a correlation between how we parsed queries and our performance on different types of queries

Arabic retrieval was especially interesting for our team, which had no personnel who could read Arabic. The lack of available translation resources left us with little alternative but to use weak machine translation systems; yet, we found bilingual performance rivaled a good monolingual baseline in terms of mean average precision (94%), had equal performance at high precision levels (such as measured in precision at 5 or 10 documents), and even achieved higher recall at 1000. Our results emphasizing language-neutral techniques indicate that excellent performance is attainable without sophisticated linguistic processing.

While we did not put significant effort into the Web track this year, we did attempt to improve our retrieval performance at high precision levels (in contrast to our previous work attempting to maximize mean average precision). We found support for several techniques currently used in the commercial sector that improve query processing efficiency without impacting high precision performance.

References

[1] A. Diekema and W-Y Hsiao, 'Cross-Language Information Retrieval Using Dutch Query Translation'. In Carol Peters (ed.) *Cross-Language Information Retrieval and Evaluation: Proceedings of the CLEF-2000 Workshop*, Lisbon, Portugal, Lecture Notes in Computer Science 2069, Springer, pp. 230-236, 2001.

[2] S. Dumais, J. Platt, D. Heckerman, M. Sahami, "Inductive Learning Algorithms and Representations for Text Categorization," in Proceedings of the 7th International Conference on Information and Knowledge Management (CIKM 98) (1998).

[3] Marti A. Hearst. Trends and controversies: Support vector machines. IEEE Intelligent Systems, 13(4):18-28, 1998.

[4] D. Hiemstra and A. de Vries, 'Relating the new language models of information retrieval to the traditional retrieval models.' CTIT Technical Report TR-CTIT-00-09, May 2000.

[5] T. Joachims, Making large-Scale SVM Learning Practical Advances in Kernel Methods - Support Vector Learning, B. Schölkopf and C. Burges and A. Smola (ed.), MIT Press, 1999.

[6] T. Joachims, "Text categorization with support vector machines: learning with many relevant features," in *Proc. 10th European Conference on Machine Learning ECML-98* (1998).

[7] W. Kraaij, 'TNO at CLEF-2001'. In Results of the CLEF-2001 Cross-Language System Evaluation Campaign (Working Notes), Darmstadt, Germany, pp. 29-40, 2001.

[8] Dave Lewis, personal communication, TREC 2001 Batch Filtering Task Experiments.

[9] B. Manjunath, P. Wu, S. Newsam, H. Shin. 'A Texture Descriptor for Browsing and Similarity Retrieval.' Journal of Signal Processing: Image Communication, 16(1), pp. 33-43, September 2000.

[10] J. Mayfield, P. McNamee, and C. Piatko, 'The JHU/APL HAIRCUT System at TREC-8.' In E. M. Voorhees and D. K. Harman, eds., *Proceedings of the Eighth Text REtrieval Conference (TREC-8),* pp. 445-451, 2000.

[11] P. McNamee, J. Mayfield, and C. Piatko, 'A Language-Independent Approach to European Text Retrieval.' In Carol Peters (ed.) *Cross-Language Information Retrieval and Evaluation: Proceedings of the CLEF-2000 Workshop*, Lisbon, Portugal, Lecture Notes in Computer Science 2069, Springer, pp. 129-139, 2001.

[12] P. McNamee, "Experiments in the retrieval of unsegmented Japanese text at the NTCIR-2 workshop," Proceedings of the 2nd NTCIR Workshop, 2001.

[13] P. McNamee and J. Mayfield, "JHU/APL experiments at CLEF 2001: Translation resources and score normalization". In Results of the CLEF-2001 Cross-Language System Evaluation Campaign (Working Notes), Darmstadt, Germany, pp. 121-132, 2001.

[14] P. McNamee, J. Mayfield, and C. Piatko, "HAIRCUT at TREC-9". In E. Voorhees and D. Harman (eds.), Proceedings of the Ninth Text REtrieval Conference (TREC-9), 2001.

[15] D. R. H. Miller, T. Leek, and R. M. Schwartz, 'A Hidden Markov Model Information Retrieval System.' In the Proceedings of the 22nd International Conference on Research and Development in

Information Retrieval (SIGIR-99), pp. 214-221, August 1999.

[16] M. Swain and D. Ballard. ‘Colour Indexing.’ International Journal of Computer Vision, 7(1), pp. 11-32, 1991.

[17] R. Zabih, J. Miller, K. Mai. ‘Feature-based Algorithms for Detecting and Classifying Scene Breaks.’ Proceedings of the 4th ACM Int. Conf. on Multimedia, 1995.

[18] Image Processing and Vision Research Lab Texture Code, http://www-iplab.ece.ucsb.edu/texture/software/index.html

[19] http://ais.gmd.de/~thorsten/svm_light/

[20] http:/english.ajeeb.com/

[21] http://www.almisbar.com/salam_trans.html

[22] Reuters Corpus, volume 1: English Language, 1996-08-20 to 1997-08-19. We gratefully acknowledge the provision of the research corpus by Reuters Limited; without it, our filtering experiments would not have been possible.