

Utility-based Information Distillation Over Temporally Sequenced Documents

Yiming Yang
Language Technologies Inst.
Carnegie Mellon University
Pittsburgh, USA
yiming@cs.cmu.edu

Abhay Harpale
Language Technologies Inst.
Carnegie Mellon University
Pittsburgh, USA
aharpale@cs.cmu.edu

Abhimanyu Lad
Language Technologies Inst.
Carnegie Mellon University
Pittsburgh, USA
alad@cs.cmu.edu

Bryan Kisiel
Language Technologies Inst.
Carnegie Mellon University
Pittsburgh, USA
bkisiel@cs.cmu.edu

Ni Lao
Language Technologies Inst.
Carnegie Mellon University
Pittsburgh, USA
nlao@cs.cmu.edu

Monica Rogati
Language Technologies Inst.
Carnegie Mellon University
Pittsburgh, USA
mrogati@cs.cmu.edu

ABSTRACT

This paper examines a new approach to information distillation over temporally ordered documents, and proposes a novel evaluation scheme for such a framework. It combines the strengths of and extends beyond conventional adaptive filtering, novelty detection and non-redundant passage ranking with respect to long-lasting information needs ('tasks' with multiple queries). Our approach supports fine-grained user feedback via highlighting of arbitrary spans of text, and leverages such information for utility optimization in adaptive settings. For our experiments, we defined hypothetical tasks based on news events in the TDT4 corpus, with multiple queries per task. Answer keys (nuggets) were generated for each query and a semi-automatic procedure was used for acquiring rules that allow automatically matching nuggets against system responses. We also propose an extension of the NDCG metric for assessing the utility of ranked passages as a combination of relevance and novelty. Our results show encouraging utility enhancements using the new approach, compared to the baseline systems without incremental learning or the novelty detection components.

Categories and Subject Descriptors

H.3.3 [Information Search and Retrieval]: Information filtering, Relevance feedback, Retrieval models, Selection process; I.5.2

General Terms

Design, Measurement, Performance, Experimentation.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

SIGIR '07, July 23–27, 2007, Amsterdam, The Netherlands.
Copyright 2007 ACM 978-1-59593-597-7/07/0007 ...\$5.00.

Keywords

Utility-based distillation, unified framework, adaptive filtering, novelty detection, passage ranking, flexible user feedback, evaluation methodology.

1. INTRODUCTION

Tracking new and relevant information from temporal data streams for users with long-lasting needs has been a challenging research topic in information retrieval. Adaptive filtering (AF) is one such task of online prediction of the relevance of each new document with respect to pre-defined topics. Based on the initial query and a few positive examples (if available), an AF system maintains a *profile* for each such topic of interest, and constantly updates it based on feedback from the user. The incremental learning nature of AF systems makes them more powerful than standard search engines that support ad-hoc retrieval (e.g. Google and Yahoo) in terms of finding relevant information with respect to long-lasting topics of interest, and more attractive for users who are willing to provide feedback to adapt the system towards their specific information needs, without having to modify their queries manually.

A variety of supervised learning algorithms (Rocchio-style classifiers, Exponential-Gaussian models, local regression and logistic regression approaches) have been studied for adaptive settings, examined with explicit and implicit relevance feedback, and evaluated with respect to utility optimization on large benchmark data collections in TREC (Text Retrieval Conferences) and TDT (Topic Detection and Tracking) forums [1, 4, 7, 15, 16, 20, 24, 23]. Regularized logistic regression [21] has been found representative for the state-of-the-art approaches, and highly efficient for frequent model adaptations over large document collections such as the TREC-10 corpus (over 800,000 documents and 84 topics). Despite substantial achievements in recent adaptive filtering research, significant problems remain unsolved regarding how to leverage user feedback effectively and efficiently. Specifically, the following issues may seriously limit the true utility of AF systems in real-world applications:

1. User has a rather 'passive' role in the conventional

adaptive filtering setup – he or she reacts to the system only when the system makes a ‘yes’ decision on a document, by confirming or rejecting that decision. A more ‘active’ alternative would be to allow the user to issue multiple queries for a topic, review a ranked list of candidate documents (or passages) per query, and provide feedback on the ranked list, thus refining their information need and requesting updated ranked lists. The latter form of user interaction has been highly effective in standard retrieval for ad hoc queries. How to deploy such a strategy for long-lasting information needs in AF settings is an open question for research.

2. The unit for receiving a relevance judgment (‘yes’ or ‘no’) is restricted to the document level in conventional AF. However, a real user may be willing to provide more informative, fine-grained feedback via highlighting some pieces of text in a retrieved document as relevant, instead of labeling the entire document as relevant. Effectively leveraging such fine-grained feedback could substantially enhance the quality of an AF system. For this, we need to enable supervised learning from labeled pieces of text of arbitrary span instead of just allowing labeled documents.
3. System-selected documents are often highly redundant. A major news event, for example, would be reported by multiple sources repeatedly for a while, making most of the information content in those articles redundant with each other. A conventional AF system would select all these redundant news stories for user feedback, wasting the user’s time while offering little gain. Clearly, techniques for novelty detection can help in principle [25, 2, 22] for improving the utility of the AF systems. However, the effectiveness of such techniques at passage level to detect novelty with respect to user’s (fine-grained) feedback and to detect redundancy in ranked lists remains to be evaluated using a measure of utility that mimics the needs of a real user.

To address the above limitations of current AF systems, we propose and examine a new approach in this paper, combining the strengths of conventional AF (incremental learning of topic models), multi-pass passage retrieval for long-lasting queries conditioned on topic, and novelty detection for removal of redundancy from user interactions with the system. We call the new process utility-based *information distillation*.

Note that conventional benchmark corpora for AF evaluations, which have relevance judgments at the document level and do not define tasks with multiple queries, are insufficient for evaluating the new approach. Therefore, we extended a benchmark corpus – the TDT4 collection of news stories and TV broadcasts – with task definitions, multiple queries per task, and answer keys per query. We have conducted our experiments on this extended TDT4 corpus and have made the additionally generated data publicly available for future comparative evaluations¹.

To automatically evaluate the system-returned arbitrary spans of text using our answer keys, we further developed an evaluation scheme with semi-automatic procedure for

acquiring rules that can match nuggets against system responses. Moreover, we propose an extension of NDCG (Normalized Discounted Cumulated Gain) [9] for assessing the utility of ranked passages as a function of both relevance and novelty.

The rest of this paper is organized as follows. Section 2 outlines the information distillation process with a concrete example. Section 3 describes the technical cores of our system called CAFÉ – CMU Adaptive Filtering Engine. Section 4 discusses issues with respect to evaluation methodology and proposes a new scheme. Section 5 describes the extended TDT4 corpus. Section 6 presents our experiments and results. Section 7 concludes the study and gives future perspectives.

2. A SAMPLE TASK

Consider a news event – the escape of seven convicts from a Texas prison in December 2000 and their capture a month later. Assuming a user were interested in this event since its early stage, the information need could be: ‘Find information about the escape of convicts from Texas prison, and information related to their recapture’. The associated lower-level questions could be:

1. How many prisoners escaped?
2. Where and when were they sighted?
3. Who are their known contacts inside and outside the prison?
4. How are they armed?
5. Do they have any vehicles?
6. What steps have been taken so far?

We call such an information need a *task*, and the associated questions as the *queries* in this task. A distillation system is supposed to monitor the incoming documents, process them chunk by chunk in a temporal order, select potentially relevant and novel passages from each chunk with respect to each query, and present a ranked list of passages to the user. Passage ranking here is based on how relevant a passage is with respect to the current query, how novel it is with respect to the current user history (of his or her interactions with the system), and how redundant it is compared to other passages with a higher rank in the list.

When presented with a list of passages, the user may provide feedback by highlighting arbitrary spans of text that he or she found relevant. These spans of text are taken as positive examples in the adaptation of the query profile, and also added to the user’s history. Passages not marked by the user are taken as negative examples. As soon as the query profile is updated, the system re-issues a search and returns another ranked list of passages where the previously seen passages are either removed or ranked low, based on user preference. For example, if the user highlights ‘...officials have posted a \$100,000 reward for their capture...’ as relevant answer to the query “What steps have been taken so far?”, then the highlighted piece is used as an additional positive training example in the adaptation of the query profile. This piece of feedback is also added to the user history as a seen example, so that in

¹URL: <http://nyc.lti.cs.cmu.edu/downloads>

future, the system will not place another passage mentioning ‘\$100,000 reward’ at the top of the ranked list. However, an article mentioning ‘...officials have doubled the reward money to \$200,000...’ might be ranked high since it is both relevant to the (updated) query profile and novel with respect to the (updated) user history. The user may modify the original queries or add a new query during the process; the query profiles will be changed accordingly. Clearly, novelty detection is very important for the utility of such a system because of the iterative search. Without novelty detection, the old relevant passages would be shown to the user repeatedly in each ranked list.

Through the above example, we can see the main properties of our new framework for utility-based information distillation over temporally ordered documents. Our framework combines and extends the power of adaptive filtering (AF), ad hoc retrieval (IR) and novelty detection (ND). Compared to standard IR, our approach has the power of incrementally learning long-term information needs and modeling a sequence of queries within a task. Compared to conventional AF, it enables a more active role of the user in refining his or her information needs and requesting new results by allowing relevance and novelty feedback via highlighting of arbitrary spans of text in passages returned by the system.

Compared to past work, this is the first evaluation of novelty detection integrated with adaptive filtering for sequenced queries that allows flexible user feedback over ranked passages. The combination of AF, IR and ND with the new extensions raises an important research question regarding evaluation methodology: how can we measure the utility of such an information distillation system? Existing metrics in standard IR, AF and ND are insufficient, and new solutions must be explored, as we will discuss in Section 4, after describing the technical cores of our system in the next section.

3. TECHNICAL CORES

The core components of CAFÉ are – 1) AF for incremental learning of query profiles, 2) IR for estimating relevance of passages with respect to query profiles, 3) ND for assessing novelty of passages with respect to user’s history, and 4) anti-redundancy component to remove redundancy from ranked lists.

3.1 Adaptive Filtering Component

We use a state-of-the-art algorithm in the field – the regularized logistic regression method which had the best results on several benchmark evaluation corpora for AF [21]. Logistic regression (LR) is a supervised learning algorithm for statistical classification. Based on a training set of labeled instances, it learns a class model which can then be used to predict the labels of unseen instances. Its performance as well as efficiency in terms of training time makes it a good candidate when frequent updates of the class model are required, as is the case in adaptive filtering, where the system must learn from each new feedback provided by the user. (See [21] and [23] for computational complexity and implementation issues).

In adaptive filtering, each query is considered as a class and the probability of a passage belonging to this class corresponds to the degree of relevance of the passage with respect to the query. For training the model, we use the

query itself as the initial positive training example of the class, and the user-highlighted pieces of text (marked as Relevant or Not-relevant) during feedback as additional training examples. To address the *cold start* issue in the early stage before any user feedback is obtained, the system uses a small sample from a retrospective corpus as the initial negative examples in the training set. The details of using logistic regression for adaptive filtering (assigning different weights to positive and negative training instances, and regularizing the objective function to prevent over-fitting on training data) are presented in [21].

The class model \vec{w}^* learned by Logistic Regression, or the query profile, is a vector whose dimensions are individual terms and whose elements are the regression coefficients, indicating how influential each term is in the query profile. The query profile is updated whenever a new piece of user feedback is received. A temporally decaying weight can be applied to each training example, as an option, to emphasize the most recent user feedback.

3.2 Passage Retrieval Component

We use standard IR techniques in this part of our system. Incoming documents are processed in chunks, where each chunk can be defined as a fixed span of time or as a fixed number of documents, as preferred by the user. For each incoming document, corpus statistics like the IDF (Inverted Document Frequency) of each term are updated. We use a state-of-the-art named entity identifier and tracker [8, 12] to identify person and location names, and merge them with co-referent named entities seen in the past. Then the documents are segmented into passages, which can be a whole document, a paragraph, a sentence, or any other continuous span of text, as preferred. Each passage is represented using a vector of TF-IDF (Term Frequency–Inverse Document Frequency) weights, where term can be a word or a named entity.

Given a query profile, i.e. the logistic regression solution \vec{w}^* as described in Section 3.1, the system computes the posterior probability of relevance for each passage \vec{x} as

$$f_{RL}(\vec{x}) \equiv P(y = 1 | \vec{x}, \vec{w}^*) = \frac{1}{(1 + e^{-\vec{w}^* \cdot \vec{x}})} \quad (1)$$

Passages are ordered by their relevance scores, and the ones with scores above a threshold (tuned on a training set) comprise the *relevance list* that is passed on to the novelty detection step.

3.3 Novelty Detection Component

CAFÉ maintains a *user history* $H(t)$, which contains all the spans of text h_i that the user highlighted (as feedback) during his or her past interactions with the system, up to the current time t . Denoting the history as

$$H(t) = \{\vec{h}_1, \vec{h}_2, \dots, \vec{h}_t\}, \quad (2)$$

the novelty score of a new candidate passage \vec{x} is computed as:

$$f_{ND}(\vec{x}) = 1 - \max_{i \in 1..t} \{\cos(\vec{x}, h_i)\} \quad (3)$$

where both candidate passage x and highlighted spans of text h_i are represented as TF-IDF vectors.

The novelty score of each passage is compared to a pre-specified threshold (also tuned on a training set), and any

passage with a score below this threshold is removed from the *relevance list*.

3.4 Anti-redundant Ranking Component

Although the novelty detection component ensures that only *novel* (previously unseen) information remains in the *relevance list*, this list might still contain the same novel information at multiple positions in the ranked list. Suppose, for example, that the user has already read about a \$100,000 reward for information about the escaped convicts. A new piece of news that the award has been increased to \$200,000 is novel since the user hasn't read about it yet. However, multiple news sources would report this news and we might end up showing (redundant) articles from all these sources in a ranked list. Hence, a ranked list should also be made non-redundant with respect to its own contents. We use a simplified version of the Maximal Marginal Relevance method [5], originally developed for combining relevance and novelty in text retrieval and summarization. Our procedure starts with the current list of passages sorted by relevance (section 3.2), filtered by Novelty Detection component (section 3.3), and generates a new non-redundant list as follows:

1. Take the top passage in the current list as the top one in the new list.
2. Add the next passage \vec{x} in the current list to the new list only if

$$f_{AR}(\vec{x}) > t$$

where

$$f_{AR}(\vec{x}) = 1 - \max_{p_i \in L_{new}} \{\cos(\vec{x}, p_i)\}$$

and L_{new} is the set of passages already selected in the new list.

3. Repeat step 2 until all the passages in the current list have been examined.

After applying the above-mentioned algorithm, each passage in the new list is sufficiently dissimilar to others, thus favoring diversity rather than redundancy in the new ranked list. The anti-redundancy threshold t is tuned on a training set.

4. EVALUATION METHODOLOGY

The approach we proposed above for information distillation raises important issues regarding evaluation methodology. Firstly, since our framework allows the output to be passages at different levels of granularity (e.g. k -sentence windows where k may vary) instead of a fixed length, it is not possible to have pre-annotated relevance judgments at all such granularity levels. Secondly, since we wish to measure the *utility* of the system output as a combination of both relevance and novelty, traditional relevance-only based measures must be replaced by measures that penalize the repetition of the same information in the system output across time. Thirdly, since the output of the system is ranked lists, we must reward those systems that present useful information (both relevant and previously unseen) using shorter ranked lists, and penalize those that present the same information using longer ranked lists. None of the existing measures in ad hoc retrieval, adaptive filtering,

novelty detection or other related areas (text summarization and question answering) have desirable properties in all the three aspects. Therefore, we must develop a new evaluation methodology.

4.1 Answer Keys

To enable the evaluation of a system whose output consists of passages of arbitrary length, we borrow the concept of *answer keys* from the Question Answering (QA) community, where systems are allowed to return arbitrary spans of text as answers. Answer keys define *what* should be present in a system response to receive credit, and are comprised of a collection of *information nuggets*, i.e. factoid units about which human assessors can make binary decisions of whether or not a system response contains them.

Defining answer keys and making the associated binary decisions are conceptual tasks that require semantic mapping [19], since system-returned passages can contain the same information expressed in many different ways. Hence, QA evaluations have relied on human assessors for the mapping between various expressions, making the process costly, time consuming, and not scalable to large query and document collections, and extensive system evaluations with various parameter settings.

4.1.1 Automating Evaluation based on Answer Keys

Automatic evaluation methods would allow for faster system building and tuning, as well as provide an objective and affordable way of comparing various systems. Recently, such methods have been proposed, more or less, based on the idea of n-gram co-occurrences. Pourpre [10] assigns a fractional recall score to a system response based on its unigram overlap with a given nugget's description. For example, a system response 'A B C' has recall 3/4 with respect to a nugget with description 'A B C D'. However, such an approach is unfair to systems that present the same information but using words other than A, B, C, and D. Another open issue is how to weight individual words in measuring the closeness of a match. For example, consider the question "How many prisoners escaped?". In the nugget 'Seven prisoners escaped from a Texas prison', there is no indication that 'seven' is the keyword, and that it must be matched to get any relevance credit. Using IDF values does not help, since 'seven' will generally not have a higher IDF than words like 'texas' and 'prison'. Also, redefining the nugget as just 'seven' does not solve the problem since now it might spuriously match any mention of 'seven' out of context. Nuggeteer [13] works on similar principles but makes binary decisions about whether a nugget is present in a given system response by tuning a threshold. However, it is also plagued by 'spurious relevance' since not all words contained in the nugget description (or known correct responses) are *central* to the nugget.

4.1.2 Nugget-Matching Rules

We propose a reliable automatic method for determining whether a snippet of text contains a given nugget, based on *nugget-matching rules*, which are generated using a semi-automatic procedure explained below. These rules are essentially Boolean queries that will only match against snippets that contain the nugget. For instance, a candidate rule for matching answers to "How many prisoners escaped?" is (Texas AND seven AND escape AND (convicts

OR prisoners)), possibly with other synonyms and variants in the rule. For a corpus of news articles, which usually follow a typical formal prose, it is fairly easy to write such simple rules to match expected answers using a bootstrap approach, as described below.

We propose a two-stage approach, inspired by Autoslog [14], that combines the strength of humans in identifying semantically equivalent expressions and the strength of the system in gathering statistical evidence from a human-annotated corpus of documents. In the first stage, human subjects annotated (using a highlighting tool) portions of on-topic documents that contained answers to each nugget². In the second stage, subjects used our rule generation tool to create rules that would match the annotations for each nugget. The tool allows users to enter a Boolean rule as a disjunction of conjunctions (e.g. ((a AND b) OR (a AND c AND d) OR (e))). Given a candidate rule, our tool uses it as a Boolean query over the entire set of on-topic documents and calculates its recall and precision with respect to the annotations that it is expected to match. Hence, the subjects can start with a simple rule and iteratively refine it until they are satisfied with its recall and precision. We observed that it was very easy for humans to improve the precision of a rule by tweaking its existing conjunctions (adding more ANDs), and improving the recall by adding more conjunctions to the disjunction (adding more ORs).

As an example, let’s try to create a rule for the nugget which says that seven prisoners escaped from the Texas prison. We start with a simple rule – (seven). When we input this into the rule generation tool, we realize that this rule matches many spurious occurrences of seven (e.g. ‘...seven states...’) and thus gets a low precision score. We can further qualify our rule – Texas AND seven AND convicts. Next, by looking at the ‘missed annotations’, we realize that some news articles mentioned “...seven prisoners escaped...”. We then replace convicts with the disjunction (convicts OR prisoners). We continue tweaking the rule in this manner until we achieve a sufficiently high recall and precision – i.e. the (small number of) misses and false alarms can be safely ignored.

Thus we can create nugget-matching rules that succinctly capture various ways of expressing a nugget, while avoiding matching incorrect (or out of context) responses. Human involvement in the rule creation process ensures high quality generic rules which can then be used to evaluate arbitrary system responses reliably.

4.2 Evaluating the Utility of a Sequence of Ranked Lists

The utility of a retrieval system can be defined as the difference between how much the user gained in terms of useful information, and how much the user lost in terms of time and energy. We calculate this utility from the utilities of individual passages as follows. After reading each passage returned by the system, the user derives some *gain* depending on the presence of relevant and novel information, and incurs a *loss* in terms of the time and energy spent in going through the passage. However, the likelihood that the user would actually read a passage depends on its position in the ranked list. Hence, for a query q , the *expected utility*

²LDC [18] already provides relevance judgments for 100 topics on the TDT4 corpus. We further ensured that these judgments are exhaustive on the entire corpus using pooling.

of a passage p_i at rank i can be defined as

$$U(p_i, q) = P(i) * (Gain(p_i, q) - Loss(p_i, q)) \quad (4)$$

where $P(i)$ is the probability that the user would go through a passage at rank i .

The expected utility for an entire ranked list of length n can be calculated simply by adding the expected utility of each passage:

$$U(q) = \sum_{i=1}^n P(i) * (Gain(p_i, q) - Loss(p_i, q)) \quad (5)$$

Note that if we ignore the loss term and define $P(i)$ as

$$P(i) \propto 1/\log_b(b + i - 1) \quad (6)$$

then we get the recently popularized metric called Discounted Cumulated Gain (DCG) [9], where $Gain(p_i, q)$ is defined as the graded relevance of passage p_i . However, without the loss term, DCG is a purely recall-oriented metric and not suitable for an adaptive filtering setting, where the system’s utility depends in part on its ability to limit the number of items shown to the user.

Although $P(i)$ could be defined based on empirical studies of user behavior, for simplicity, we use $P(i)$ exactly as defined in equation 6.

The gain $G(p_i, q)$ of passage p_i with respect to the query q is a function of – 1) the number of relevant nuggets present in p_i , and 2) the novelty of each of these nuggets. We combine these two factors as follows. For each nugget N_j , we assign an initial weight w_j , and also keep a count n_j of the number of times this nugget has been seen by the user in the past. The gain derived from each subsequent occurrence of the same nugget is assumed to reduce by a dampening factor γ . Thus, $G(p_i, q)$ is defined as

$$G(p_i, q) = \sum_{N_j \in C(p_i, q)} w_j * \gamma^{n_j} \quad (7)$$

where $C(p_i, q)$ is the set of all nuggets that appear in passage p_i and also belong to the answer key of query q . The initial weights w_j are all set of be 1.0 in our experiments, but can also be set based on a pyramid approach [11]. The choice of dampening factor γ determines the user’s tolerance for redundancy. When $\gamma = 0$, a nugget will only receive credit for its first occurrence i.e. when n_j is zero³. For $0 < \gamma < 1$, a nugget receives smaller credit for each successive occurrence. When $\gamma = 1$, no dampening occurs and repeated occurrences of a nugget receive the same credit. Note that the nugget occurrence counts are preserved between evaluation of successive ranked lists returned by the system, since the users are expected to remember what the system showed them in the past.

We define the loss $L(p_i, q)$ as a constant cost c (we use 0.1) incurred when reading a system-returned passage. Thus, our metric can be re-written as

$$U(q) = \sum_{i=1}^n \frac{Gain(p_i, q)}{\log_b(b + i - 1)} - L(n) \quad (8)$$

where $L(n)$ is the loss associated with a ranked list of length n :

$$L(n) = c \cdot \sum_{i=1}^n \frac{1}{\log_b(b + i - 1)} \quad (9)$$

³Note that $0^0 = 1$

Due to the similarity with Discounted Cumulated Gain (DCG), we call our metric Discounted Cumulated Utility (DCU). The DCU score obtained by the system is converted to a Normalized DCU (NDCU) score by dividing it by the DCU score of the ideal ranked list, which is created by ordering passages by their decreasing utility scores $U(p_i, q)$ and stopping when $U(p_i, q) \leq 0$ i.e. when the gain is less than or equal to the cost of reading the passage.

5. DATA

TDT4 was the benchmark corpus used in TDT2002 and TDT2003 evaluations. The corpus consists of over 90,000 news articles from multiple sources (AP, NYT, CNN, ABC, NBC, MSNBC, Xinhua, Zaobao, Voice of America, PRI the World, etc.) published between October 2000 and January 2001, in the languages of Arabic, English, and Mandarin. Speech-recognized and machine-translated versions of the non-English articles were provided as well.

LDC [18] has annotated the corpus with 100 topics, that correspond to various news events in this time period. Out of these, we selected a subset of 12 actionable events, and defined corresponding tasks for them⁴. For each task, we manually defined a profile consisting of an initial set of (5 to 10) queries, a free-text description of the user history, i.e., what the user already knows about the event, and a list of known on-topic and off-topic documents (if available) as training examples.

For each query, we generated answer keys and corresponding nugget matching rules using the procedure described in section 4.1.2, and produced a total of 120 queries, with an average of 7 nuggets per query.

6. EXPERIMENTS AND RESULTS

6.1 Baselines

We used Indri [17], a popular language-model based retrieval engine, as a baseline for comparison with CAFÉ. Indri supports standard search engine functionality, including pseudo-relevance feedback (PRF) [3, 6], and is representative of a typical query-based retrieval system. Indri does not support any kind of novelty detection.

We compare Indri with PRF turned on and off, against CAFÉ with user feedback, novelty detection and anti-redundant ranking turned on and off.

6.2 Experimental Setup

We divided the TDT4 corpus spanning 4 months into 10 *chunks*, each defined as a period of 12 consecutive days. At any given point of time in the distillation process, each system accessed the past data up to the current point, and returned a ranked list of up to 50 passages per query.

The 12 tasks defined on the corpus were divided into a training and test set with 6 tasks each. Each system was allowed to use the training set to tune its parameters for optimizing NDCU (equation 8), including the relevance threshold for both Indri and CAFÉ, and the novelty and anti-redundancy thresholds for CAFÉ.

The NDCU for each system run is calculated automatically. User feedback was also simulated – relevance judgments for each system-returned passage (as determined by the nugget matching rules described in section 4.1.2) were

⁴URL: <http://nyc.lti.cs.cmu.edu/downloads>

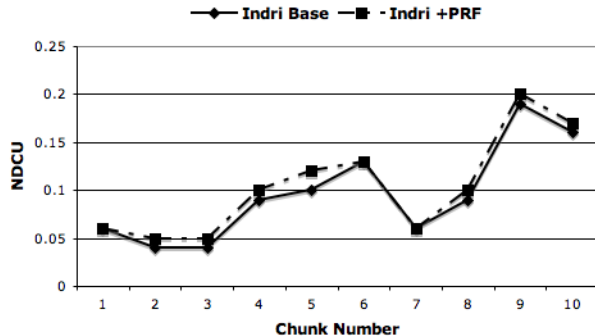


Figure 1: Performance of Indri across chunks

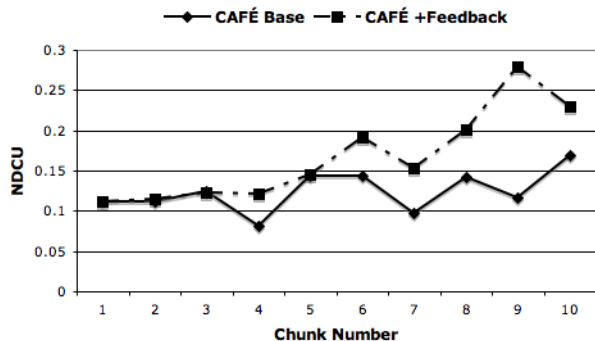


Figure 2: Performance of CAFÉ across chunks

used as user feedback in the adaptation of query profiles and user histories.

6.3 Results

In Table 1, we show the NDCU scores of the two systems under various settings. These scores are averaged over the six tasks in the test set, and are calculated with two dampening factors (see section 4.2): $\gamma = 0$ and 0.1, to simulate no tolerance and small tolerance for redundancy, respectively.

Using $\gamma = 0$ creates a much more strict metric since it does not give any credit to a passage that contains relevant but redundant information. Hence, the improvement obtained from enabling user feedback is smaller with $\gamma = 0$ than the improvement obtained from feedback with $\gamma = 0.1$. This reveals a shortcoming of contemporary retrieval systems – when the user gives positive feedback on a passage, the systems gives higher weights to the terms present in that passage and tends to retrieve other passages containing the same terms – and thus – usually the same information. However, the user does not benefit from seeing such redundant passages, and is usually interested in other passages containing related information. It is informative to evaluate retrieval systems using our utility measure (with $\gamma = 0$) which accounts for novelty and thus gives a more realistic picture of how well a system can generalize from user feedback, rather than using traditional IR measures like recall and precision which give an incomplete picture of improvement obtained from user feedback.

Sometimes, however, users might indeed be interested in seeing the same information from multiple sources, as an

Table 1: NDCU Scores of Indri and CAFÉ for two dampening factors (γ), and various settings (F: Feedback, N: Novelty Detection, A: Anti-Redundant Ranking)

γ	Indri		CAFÉ				
	Base	+PRF	Base	+F	+F+N	+F+A	+F+N+A
0	0.19	0.19	0.22	0.23	0.24	0.24	0.24
0.1	0.28	0.29	0.24	0.35	0.35	0.36	0.36

indicator of its importance or reliability. In such a case, we can simply choose a higher value for γ which corresponds to a higher tolerance for redundancy, and hence let the system tune its parameters accordingly.

Since documents were processed chunk by chunk, it would be interesting to see how the performance of systems improves over time. Figures 1 and 2 show the performance trends for both the systems across chunks. While the performance with and without feedback on the first few chunks is expected to be close, for subsequent chunks, the performance curve with feedback enabled rises above the one with the no-feedback setting. The performance trends are not consistent across all chunks because on-topic documents are not uniformly distributed over all the chunks, making some queries ‘easier’ than others in certain chunks. Moreover, since Indri uses pseudo-relevance feedback while CAFÉ uses feedback based on actual relevance judgments, the improvement in case of Indri is less dramatic than that of CAFÉ.

7. CONCLUDING REMARKS

This paper presents the first investigation on utility-based information distillation with a system that learns the long-lasting information needs from fine-grained user feedback over a sequence of ranked passages. Our system, called CAFÉ, combines adaptive filtering, novelty detection and anti-redundant passage ranking in a unified framework for utility optimization. We developed a new scheme for automated evaluation and feedback based on a semi-automatic procedure for acquiring rules that allow automatically matching nuggets against system responses. We also proposed an extension of the NDCG metric for assessing the utility of ranked passages as a weighted combination of relevance and novelty. Our experiments on the newly annotated TDT4 benchmark corpus show encouraging utility enhancement over Indri, and also over our own system with incremental learning and novelty detection turned off.

8. ACKNOWLEDGMENTS

We would like to thank Rosta Farzan, Jonathan Grady, Jaewook Ahn, Yefei Peng, and the Qualitative Data Analysis Program at the University of Pittsburgh lead by Dr. Stuart Shulman for their help with collecting and processing the extended TDT4 annotations used in our experiments. This work is supported in parts by the National Science Foundation (NSF) under grant IIS-0434035, and the Defense Advanced Research Project Agency (DARPA) under contracts NBCHD030010 and W0550432. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the sponsors.

9. ADDITIONAL AUTHORS

Jian Zhang (jianzhan@stat.purdue.edu)*, Jaime Carbonell (jgc@cs.cmu.edu)[†], Peter Brusilovsky (peterb+@pitt.edu)[‡], Daqing He(dah44@pitt.edu)[‡]

10. REFERENCES

- [1] J. Allan. Incremental Relevance Feedback for Information Filtering. *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 270–278, 1996.
- [2] J. Allan, C. Wade, and A. Bolivar. Retrieval and Novelty Detection at the Sentence Level. *Proceedings of the ACM SIGIR conference on research and development in information retrieval*, 2003.
- [3] C. Buckley, G. Salton, and J. Allan. Automatic Retrieval with Locality Information using SMART. *NIST special publication*, (500207):59–72, 1993.
- [4] J. Callan. Learning While Filtering Documents. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 224–231, 1998.
- [5] J. Carbonell and J. Goldstein. The use of MMR, Diversity-based Reranking for Reordering Documents and Producing Summaries. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 335–336, 1998.
- [6] E. Efthimiadis. Query Expansion. *Annual Review of Information Science and Technology (ARIST)*, 31:p121–87, 1996.
- [7] J. Fiscus and G. Duddington. Topic Detection and Tracking Overview. *Topic Detection and Tracking: Event-based Information Organization*, pages 17–31.
- [8] R. Florian, H. Hassan, A. Ittycheriah, H. Jing, N. Kambhatla, X. Luo, N. Nicolov, and S. Roukos. A Statistical Model for Multilingual Entity Detection and Tracking. *NAACL/HLT*, 2004.
- [9] K. Järvelin and J. Kekäläinen. Cumulated Gain-based Evaluation of IR Techniques. *ACM Transactions on Information Systems (TOIS)*, 20(4):422–446, 2002.
- [10] J. Lin and D. Demner-Fushman. Automatically Evaluating Answers to Definition Questions. *Proceedings of the 2005 Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing (HLT/EMNLP 2005)*, 2005.

*Statistics Dept., Purdue University, West Lafayette, USA

[†]Language Technologies Inst., Carnegie Mellon University, Pittsburgh, USA

[‡]School of Information Sciences, Univ. of Pittsburgh, Pittsburgh, USA

- [11] J. Lin and D. Demner-Fushman. Will Pyramids Built of nUggets Topple Over. *Proceedings of HLT-NAACL*, 2006.
- [12] X. Luo, A. Ittycheriah, H. Jing, N. Kambhatla, and S. Roukos. A Mention-synchronous Coreference Resolution Algorithm based on the Bell Tree. *Proc. of ACL*, 4:136–143, 2004.
- [13] G. Marton. Nuggeteer: Automatic Nugget-Based Evaluation Using Descriptions and Judgments. *HLT/NAACL*, 2006.
- [14] E. Riloff. Automatically Constructing a Dictionary for Information Extraction Tasks. *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 811–816, 1993.
- [15] S. Robertson and S. Walker. Microsoft Cambridge at TREC-9: Filtering track. *The Ninth Text REtrieval Conference (TREC-9)*, pages 361–368.
- [16] R. Schapire, Y. Singer, and A. Singhal. Boosting and Rocchio Applied to Text Filtering. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, pages 215–223, 1998.
- [17] T. Strohman, D. Metzler, H. Turtle, and W. Croft. Indri: A Language Model-based Search Engine for Complex Queries. *Proceedings of the International Conference on Intelligence Analysis*, 2004.
- [18] The Linguistic Data Consortium.
<http://www ldc.upenn.edu/>.
- [19] E. Voorhees. Overview of the TREC 2003 Question Answering Track. *Proceedings of the Twelfth Text REtrieval Conference (TREC 2003)*, 2003.
- [20] Y. Yang and B. Kisiel. Margin-based Local Regression for Adaptive Filtering. *Proceedings of the twelfth international conference on Information and knowledge management*, pages 191–198, 2003.
- [21] Y. Yang, S. Yoo, J. Zhang, and B. Kisiel. Robustness of Adaptive Filtering Methods in a Cross-benchmark Evaluation. *Proceedings of the 28th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 98–105, 2005.
- [22] C. Zhai, W. Cohen, and J. Lafferty. Beyond Independent Relevance: Methods and Evaluation Metrics for Subtopic Retrieval. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 10–17, 2003.
- [23] J. Zhang and Y. Yang. Robustness of Regularized Linear Classification Methods in Text Categorization. *Proceedings of the 26th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 190–197, 2003.
- [24] Y. Zhang. Using Bayesian Priors to Combine Classifiers for Adaptive Filtering. *Proceedings of the 27th annual international conference on Research and development in information retrieval*, pages 345–352, 2004.
- [25] Y. Zhang, J. Callan, and T. Minka. Novelty and Redundancy Detection in Adaptive Filtering. *Proceedings of the 25th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, 2002.