Finding New Information Via Robust Entity Detection

Francisco Iacobelli and **Nathan Nichols** and **Larry Birnbaum** and **Kristian Hammond**
Intelligent Information Laboratory
Northwestern University
2133 Sheridan Rd.
Evanston, IL 60208, U.S.
f-iacobelli@u.northwestern.edu; ndnichols@gmail.com {hammond;birnbaum}@cs.northwestern.edu

Abstract

Journalists and editors work under pressure to collect relevant details and background information about specific events. They spend a significant amount of time sifting through documents and finding new information such as facts, opinions or stakeholders (i.e. people, places and organizations that have a stake in the news). Spotting them is a tedious and cognitively intense process. One task, essential to this process, is to find and keep track of stakeholders. This task is taxing cognitively and in terms of memory. Tell Me More offers an automatic aid to this task. Tell Me More is a system that, given a seed story, mines the web for similar stories reported by different sources and selects only those stories which offer new information with respect to that original seed story. Much like a journalist, the task of detecting named entities is central to its success. In this paper we briefly describe Tell Me More and, in particular, we focus on Tell Me More's entity detection component. We describe an approach that combines off-the-shelf named entity recognizers (NERs) with WPED, an in-house publicly available NER that uses Wikipedia as its knowledge base. We show significant increase in precision scores with respect to traditional NERs. Lastly, we present an overall evaluation of Tell Me More using this approach.

Introduction

Journalists and editors work under tight deadlines and are forced to gather as much background and details as they can about a particular situation or event. They have to keep track of useful written sources and they have to be able to record what aspects and what portions of the sources provided useful information.

Often times their work requires reading, synthesizing, contrasting and comparing the narratives of the same event from different sources. They look for new information among documents. They focus, among other things, in finding new facts, testimonials and mentions of stakeholders, such as people, places, companies, organizations, etc.

Because the internet has made available vasts amounts of content about virtually anything, the journalist's task of separating the relevant from the irrelevant becomes demanding in terms of cognitive load and memory use. In particular, keeping track of all the stakeholders they find and the different ways in which they can be referred to (e.g. "North Atlantic Alliance," "NATO," "North Atlantic Treaty Organization") and learning new ones along the way are tasks that exhibit decreasing performance with each additional document that is analyzed (cf. Keppel and Underwood 1962).

Tell Me More (Iacobelli, Birnbaum, and Hammond 2010) is a system that, given a seed story, augments it by finding new information reported by other sources. It does this by mining the web for stories that are similar to the seed story and by comparing them to that seed story using a number of metrics. It then presents only those paragraphs where stories differ from each other, organized by categories that make visible the criteria used to select them. Tell Me More has been well received by a number of journalists, editors and media companies who have tested it.

Recognizing new information, in general, is a computationally difficult task. To make this problem tractable Tell Me More focuses on finding instances of numbers, quotes and entities (people, places and organizations) that can be reliable recognized and identified as new information. Because named entity recognizers (NERs) have a few drawbacks for this task, such as poor consolidation of entities that are referred to in different ways, it is of central importance to have a robust approach to named entity recognition.

In this paper we present a brief overview of Tell Me More, a detailed description and evaluation of its named entity recognition algorithms and an overall evaluation of the algorithms that determine what constitutes new information. We finish with conclusions and future steps.

Architecture

Tell Me More employs five core modules to extract and present new information to the user. What follows is a quick and slightly updated summary of each module's task. More detail on each of these modules can be found in (Iacobelli, Birnbaum, and Hammond 2010).

1. **Content Gathering:** This module gathers stories that are similar to the seed story. It does so by developing a basic model of the source article through text analytics (see module 3 below) and uses this to form a query that is sent to a search engine (at this point the system can query Google, Google News and Yahoo News). If there are no results, this module employs several strategies to reformulate the queries to maximize the likelihood of obtaining

relevant results (Budzik, Hammond, and Birnbaum 2001; Iacobelli, Birnbaum, and Hammond 2010).

2. **Content Filtering:** This module filters results and eliminates documents that are exactly the same as the seed document or documents that are too different using a TF*IDF (Salton and Buckley 1988) vector representations of each paragraph. Content filtering also takes care of discarding articles that look like compendia of articles or spam using simple heuristics.

3. **Text Analytics:** This module is used to develop a statistical and heuristic model of each filtered document. For every document that it examines, it extracts: (a) quotes; (b) quantifiers for tangible things as denoted by the presence of plural noun phrases in the sentences where numbers appear; (c) a TF*IDF vector representation of each paragraph; and (d) the names of people, places and organizations using a boosting approach to named entity recognition.

4. **Difference Metrics** After the text analytics for each new paragraph are gathered, this module compares them with the text analytics of the previously seen text and determines which paragraphs contain new information. Each feature has specialized algorithms for comparison. These comparisons are tracked by the difference metrics and the dimensions of the differences are determined.

5. **Presentation** Research suggests that clustering search results in sensible categories affords easier navigation (Käki 2005). Our own experiments on categorization of new information strongly suggest the same (Iacobelli, Birnbaum, and Hammond 2010). Thus, the Presentation module categorizes and ranks the new information based on the output of the Difference Metrics module. To make the selection criteria of new information visible to users, the system recommends paragraphs in one of four sections based on the difference metrics obtained earlier: (a)"additional names:" new proper names, countries, cities and organizations not present in the seed story; (b) "new numbers:" any quantifier not appearing in the seed story; (c) "additional quotes:" quotes not appearing in the seed story and (d) "supplementary information:" paragraphs that are sufficiently dissimilar from any previously seen paragraphs, as measured by the cosine of their vector representations.

While each of these modules is important for Tell Me More, its core tasks are to obtain a robust representation of the news stories and a reliable metric to determine differences among documents. Therefore, its most important modules are the *Text Analytics* and *Difference Metrics* modules.

Named entity recognition is a central part of Tell Me More's ability to create a representation of a news story. Currently, the system uses a boosting approach to named entity recognition. Tell Me More uses an off-the-shelf NER and it boosts its performance using the Wikipedia Entity Detector (WPED), an in-house, publicly available NER that performs very well in hard cases, such as consolidating popular entities that are referred to by different names.

In the following sections we provide background on the challenges presented by automatic named entity recognition, we describe our approach to overcome some of them and we present an evaluation of the performance of the Difference Metrics module.

Background: Named Entity Recognition

People can recognize the names of places, organizations and other entities in text easily. Moreover, with some background knowledge, or judging from the text, they can link different names as referring to the same entity. For example, "Bill Clinton" and "President Clinton" or "GOP" and "Republican Party."

However, to annotate named entities automatically, NERs have to overcome some challenges. First, NERs need to differentiate between multi-word entities and many consecutive entities. For example "Standard and Poor" is one entity, but "Toyota and Ford" are two. Similarly, in the name "Sir Elton John", the "Sir" can be dropped, but in "Mr. Bean," the suffix "Mr." is part of the named entity. NERs have tried to overcome this problem by either including rule based heuristics (Etzioni et al. 2005) or by gathering statistics from large corpora. One such NER (Downey, Broadhead, and Etzioni 2007), attempts to solve this by accumulating n-grams statistics on a web corpus and using capitalization cues. This simple method called LEX++ outperformed more complex statistical models such as conditional random fields (Lafferty, McCallum, and Pereira 2001). It also outperformed the entity detection module of KnowItAll, an unsupervised rule based system for fact collection (Etzioni et al. 2005). However, the cases where LEX++ failed were those where distinct entities were often mentioned together. For example "Intel and AMD." These are precisely the cases where our approach, presented in the next section, shows improvements.

A second, problem has to do with entities that do not follow basic rules of capitalization. Some organizations and locations are not often capitalized entirely, thus making it hard for the NER to recognize the entire entity. Such is the case of "Gulag archipelago," for example. Semi supervised methods, such as Li and McCallum (2005), that group content words leveraging *part of speech* tags may improve the detection of such entities.

Lastly, NERs have trouble consolidating entities. For example: the text *"Farrah Fawcett, when married to Lee Majors, was known as Farrah Fawcett-Majors"* contains two entities: Farrah Fawcett and Lee Majors. however, most entity detection systems will also expose one additional false positive: *Farrah Fawcett-Majors*. Some systems, although not NERs per se, can deal with many of these cases when the entities are well known. For example, Yerva, Miklós, and Aberer (2010) use several features derived from internet searches to establish a similarity between two names to see if they are actually the same entity. Their work, however, has only been tested on people's names. Another, more general approach, is the case of wikifiers (e.g. Milne and Witten 2008), which leverage the knowledge of the internet community by using Wikipedia as the main dictionary. Wikifiers, however, do not have an ontology beyond the tags specified by Wikipedia users. Moreover, these tags are largely inconsistent and significant work needs to be done to be able to correctly identify people, places and organizations.

In the following sections we present a successful approach that uses OpenCalais, a freely available NER and *boosts* its precision with a proprietary NER developed in our lab: The Wikipedia Entity Detector (WPED). WPED queries Wikipedia and is supported by a set of heuristics to improve precision. Additionally, it builds an ontology on top of Wikipedia entries, making it more useful for entity detection than other Wikipedia based NERs.

Wikipedia Entity Detector

The Wikipedia Entity Detector (WPED) consists of two main components: (a) a WPED Builder that prepares the data to be queried efficiently; and (b) a web service that detects entities by reading text and querying the data for potential entities.

WPED Builder

Building WPED comprises two distinct phases. First, the WPED builder parses an XML dump of Wikipedia [1] and stores it in a MySQL database. The system stores basic information about each "entry" (i.e. page in Wikipedia). This information includes: (a) the name of the entry; (b) other entries that link to it; (c) the categories the entry is in; (d) alternate references (names) for a specific entry pulled from Wikipedia's explicit #REDIRECT pages and #DISAMBIGUATION pages; and (e) how often the entry title is capitalized in its own page. Because all entry names must begin with a capital letter, this last feature allows us to determine whether the entry is commonly spelled with an initial capital or lower case letter.

Second, once the XML is parsed, the builder structures this information in memory and allows clients to issue requests. At the core of the system is a large *trie* data structure (Fredkin 1960) that maps references and titles to list of possible entities referred to by that title or reference. For example, both "Bush" and "George W. Bush" map to a list of entries that include "George W. Bush," "Bush" the British rock band, the last name, etc. When inserting entries into the *trie*, the system may also choose to adjust word case based on their frequency of capitalization within their entry. For example, "Barack Obama" is always capitalized in his own article, but "police" is typically lowercase in its article. Therefore, when "police" is stored in the *trie*, it will be saved in its lowercase form.

Although the *trie* works essentially like a dictionary, mapping entity names to a list of values, its short access time allow us to continuously scan the text for these names. This relieves us from relying on imperfect orthographic and lexical rules to suggest candidate entities.

Detection

The actual detection algorithm works as follows. The *trie* data structure organizes text by forming a root node with its branches being all letters that may start an entity. Nodes that mark the final letter of an entity contain pointers to the different disambiguations for that entity. When text is given to WPED, it begins matching characters in the *trie*; it continues consuming characters until there is no longer a match. If there are candidate entities in the *trie* at a node and it coincides with the end of a word in the text, an entity is considered to be detected. The cursor is then advanced one word in the text, and the process is repeated. For example, Figure 1 shows the word "Fiat" represented as the sub-graph $F \rightarrow I \rightarrow A \rightarrow T$ where "T" is the leaf node. When the system encounters this sequence of letters in the text, and if after the "T" the system finds a space in the text, it will assume an entity boundary because there is no space character as a node of "T." It, then, checks to see if the last accessed node contains a pointer to the entity "Fiat," the auto maker. If it does, it recognizes the entity. Otherwise, the search process starts over with the next character.

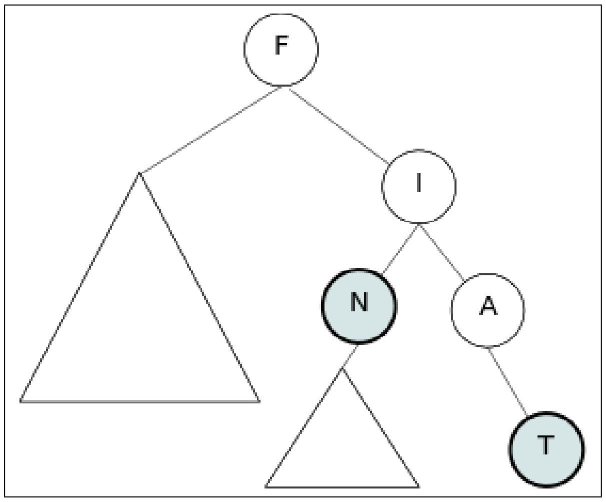

Figure 1: Trie Data Structure. The words FIAT and FIN are represented. The dark nodes denote the boundaries of words and are mapped to a dictionary. Triangles denote other subtrees

Once the entire text has been processed in this way, WPED has a dictionary mapping the detected surface forms to a list of possible entities. This dictionary is called a *ResultSet*.

Disambiguation

At this point, WPED may have encountered several candidate entities for a given word (e.g. the word "Bush" maps to the former presidents of the U.S., a band, etc.). To determine the most likely entity, WPED currently has two disambiguation strategies. The first, and simplest strategy is based on *popularity*; that is, the candidate entry with the most incoming links from other Wikipedia articles is chosen as the most likely entity. This strategy can incorrectly identify some "easy" entities; for example: in the text *"George Bush was President from 1988-1992 and his vice president was Dan Quayle"*, the popularity strategy maps *"George Bush"* to

[1] `http://en.wikipedia.org/wiki/Wikipedia_database`

his son, *George W. Bush*. However, popularity can be computed very quickly and, more importantly, it performs fairly well in practice, especially in the news domain. We present an evaluation of this strategy later in the paper.

The second strategy, based on *proximity*, is slower but more sound than the popularity strategy. The proximity strategy exploits the idea that entities mentioned in the same text are likely to be related; the "George Bush" that is mentioned near "Dan Quayle" is likely to be the George Bush that worked with Dan Quayle, not his son. To determine how closely related different Wikipedia entries are, we define a distance metric between any two entries. Specifically, the distance from entry A to entry B is $\frac{A \rightarrow B}{A_{total}}$ where $A \rightarrow B$ is the number of links from entry A to B and A_{total} is the total number of outgoing links from entry A. Bigger numbers indicate a better likelihood of being the target entity.

Unfortunately, WPED can detect everything that has an entry in Wikipedia; this includes nearly every date, thousands of common nouns, every first name, etc. To get rid of these non-helpful "entities", we have hand-coded a few rules that eliminate the false positives. For example, if an entity has a lowercase first letter, and the only surface form used is a common dictionary word, that entity is discarded. In practice, these are heuristics that improve WPED for our purposes.

Classifications and Meta Information

For most applications that use NERs it is essential to know the type of the entities detected. Although Wikipedia data is categorized by users, this categorization can be frustratingly inconsistent because it is not arranged in any kind of ontology. To enable this sort of classification, we manually built a higher-level classification system on top of Wikipedia's categories. The system allows clients to create their own ontologies adding regular expressions that match user's categories and apply classifications. Ontologies in WPED are trees that can have any number of levels and nodes. For example, any Wikipedia entry that is in a category whose title ends with "lawyers" (*American lawyers*, *European lawyers*, etc.) is classified as a *Lawyer* and any category whose title ends with "politicians" is a *politician* At a higher level, one can specify that every *Lawyer* and every *politician* is also a *Person*.

After recording the classifications, WPED has a final list of entities detected in the text along with meta information about them. However, before it returns entities to the client it performs a few small final tasks. First, it groups all the terms found in the text that could refer to a single entity, and associates them to that entity along with counts of each term. The list of entities is then converted into XML and then returned to the client.

Supplementing WPED with off the shelf entity detection

Because WPED can only detect entities that are present in Wikipedia, it obviously misses some entities; examples include names of regular people or officials that are not very well known, and small company names such as a local pub or bank.

A NER based on machine learning techniques and heuristics is better suited for these types of detection. Instead of over-complicating the implementation of WPED with such functionality we decided to combine the power of one such NER, OpenCalais[2], with WPED.

OpenCalais, by Thomson-Reuters, is a freely available NER, and we use it to detect an initial set of entities from text. The same text is fed then to WPED. For each entity detected, OpenCalais and WPED store all the terms in the text that refer to that entity. These are called instances. For example, the entity "George W. Bush" may have been referred to by two instances "G. W. Bush" and "President Bush." After the entities have been detected, all instances of entities detected by WPED are compared to those detected by OpenCalais. If an entity detected by OpenCalais corresponds to an instance of a WPED entity, then the entity detected by OpenCalais is discarded and the parent entity detected by WPED is added to the resulting set of entities. If an entity is detected by OpenCalais and it does not correspond to any entities detected by WPED, then it is added as-is to the resulting set of entities. The meta information associated with entities detected by both WPED and OpenCalais is equal to the WPED ontology for that entity. For all other entities, the OpenCalais ontology is kept.

Evaluation

To evaluate this approach we conducted a comparison of entity detection using WPED (using the popularity disambiguation strategy), OpenCalais and Wikipedia Miner Wikifier (WMW) (Milne and Witten 2008), a robust wikipedia NER that detect entities by combining machine learning algorithms with a Wikipedia dictionary. We used thirteen texts taken from random news stories. The texts were annotated with the names or people, places and organizations. A total of 227 named entities were tagged to set the "ground truth." Precision and recall was computed for each NER based on the metadata provided for each entity.

Precision, however, is the most important of these measures for Tell Me More. On the one hand, false positives harm its performance because they mislead the user therefore, further complicating the task. On the other hand, because our interest is to detect entities in online content, which can be vast, even lower recall percentages across many documents will still produce satisfactory output. For these reasons, we will pay special attention to the comparisons of precisions scores.

A statistical comparison of precision was performed between OpenCalais and our boosted approach (OpenCalais + WPED). Because results for precision are skewed (i.e. they are very high) the distribution of the data is not normal. Because our approach depends on OpenCalais, the data is not independent. Therefore, we used a paired Wilcoxon signed rank test to compare precision scores of the 13 articles. The test showed significant improvements of our approach at the $p < 0.05$ level.

[2]`http://www.opencalais.com`

	OC+WPED	OC	WPED	WMW
Precision	0.95*	0.91	0.81	-
Recall	0.70	0.69	0.42	0.42
F-measure	0.80	0.79	0.55	-

Table 1: Precision, recall and F scores for OpenCalais plus WPED (OC+WPED), OpenCalais (OC), WPED and Wikipedia Miner (WMW). * Significantly different from OC alone; $p < 0.05$

Results As Table 1 shows, a combined approach results in a more precise named entity recognition. Looking closely at the data, we realized that WPED improved OpenCalais performance exactly where instances of the same entity were not obvious. For example, the "Legal Defense and Educational Fund" and the "NAACP Legal Defense and Educational Fund" are the same entity. OpenCalais detected two entities while WPED detected one entity with two instances. Because the Wikipedia Miner Wikifier (WMW) does not provide a comparable ontology for each entity, the table only reports its recall measures on the dataset. The recall rate of WMW was comparable to WPED's which uses simpler algorithms. Lastly, the low recall scores WPED and a well established Wikifier underline the complexity of the task.

Evaluation of Tell Me More Difference Metrics

Preliminary user evaluations suggest that Tell Me More is a viable system, well organized and able to provide relevant background information and detail about news stories (Iacobelli, Birnbaum, and Hammond 2010).

In this section we focus on measuring precision and recall of its Difference Metric module, which discovers new actors (people, places and organizations), numbers and quotes. To establish a gold standard we built a small corpus of 13 different news stories, selected at random from Google news, that were covered by at least three sources each. For each set of three news stories, one random story was designated as a "seed" story. Then, we randomly chose two other versions of the same story to incrementally find paragraphs with new information that the seed story did not report. A human coder read the seed story and the additional news stories sequentially and annotated actors, numbers and quotes that were new with respect to previously read text.

After the corpus was tagged, we, proceeded to compare the Tell Me More's performance on the same corpus. We computed precision, recall and two F scores, explained in the next section, for stakeholders, numbers and quotes and one other category of new information: "long quotes."

Long quotes are a subset of all quotes. They have five or more words. The reason for this distinction is that we find, in practice, that they tend to provide better evidence of people's opinions as opposed to terms or short expressions which provide only partial evidence of those opinions. For example, contrast the text of one news source: *(...) and said there was still "a path available"* with the full quote reported on a different source: *"We have provided a path whereby [some country] can reach out to the international community, engage, and become a part of international norms," Mr. Obama said. "What we've been seeing over the last several days, the last couple of weeks, obviously is not encouraging in terms of the path."* It is clear that the longer quote provides a more accurate depiction of Mr. Obama's thoughts, while the shorter quote *"a path available"* serves to soften these remarks.

Results and Discussion In addition to precision and recall, we computed two F scores. The traditional F_1 score and a weigthed F_β score. F_1 weights precision and recall equally. However, as we mentioned earlier, Tell Me More's success is more concerned with precision than recall. Therefore, we used the weighted F_β measure as in Schiffman and Mckeown(2004) to favor precision.

From Table 2 we can see that our F_β scores are very high for all categories and, in fact, this is due mainly to precision scores above 80% percent. Recall around 60% in most categories is due to common hard problems in NLP such as co-reference resolution in the case of names and some numbers, and correct part of speech (POS) tagging in the case of numbers. For example, the POS tagger used did not considered the word "million" to be a noun and, because such tagging is essential in the detection of quantifiers, the system failed to detect many money amounts. The system also ignored spelled out numbers. F_β for longer quotes (more than 5 words) is very high, 0.91. Longer quotes are a meaningful distinction because they are usually the ones bearing more information. This high precision results in the retrieval of genuinely new and more comprehensive opinions. Perhaps this explains, to an extent, why previous research suggested that the new quotes category was well liked by users (Iacobelli, Birnbaum, and Hammond 2010).

Category	Precision	Recall	F_1	F_β
New Actors	0.83	0.58	0.68	0.79
New Numbers	0.81	0.55	0.66	0.77
New Quotes	0.93	0.57	0.71	0.87
New "long" Quotes	0.95	0.69	0.8	0.91

Table 2: Standard IR Scores in five categories.

Related Work

Significant research on detecting new information has been conducted at the TREC novelty track competitions. In this task, the highest precision and recall measures were around 0.55 and 0.78.(Soboroff and Harman 2003).

New information detection research has largely been used for news summarization software (Sweeney, Crestani, and Losada 2008; Schiffman 2005), however, other approaches include augmenting news stories with information on particular people and events. For example Wagner et al. (2009) mined the web for stories of kidnappings and used situation frames to store information. This allowed users to get augmented information on kidnapping stories such as timelines and people involved. NewsJunkie (Gabrilovich, Dumais, and Horvitz 2004) utilizes vector representations and entity detection to judge novel content in news. This is used to provide readers with updates, developments and recaps of news stories. Newsjunkie, however, does not specify what exactly is new information in the articles presented. A similar problem is present in Park, Lee, and Song (2010), where the system attempts to cluster stories about the same event based on points of view. However, the clusters are not labeled and the system is unable to provide meta information about them. In contrast, Tell Me More not only detects new information, but presents it in a way that makes the selection criteria visible to the user.

Conclusion and Future Work

Tell Me More is a news reading system that displays items of new information alongside with a news story. These items are categorized in a way that makes the editorial decisions of the recommendation clear to the user. Tell Me More relies on sound algorithms to create a representation of the news stories it processes. Named entity detection is, therefore, crucial to its success.

In this paper we have presented Tell Me More's entity detection approach and showed that it results in improved recognition when compared to state of the art off-the-shelf NERs. In particular, this approach improves precision when there is a need to consolidate different instances of the same entity and when entities appear in lower case. The contributions of our approach are twofold: (a) it offers an ontology that allows programmers to "make sense" of the entities detected; and (b) WPED and our boosting approach are easy to replicate due to the simplicity of its algorithms and use of high performance off-the-shelf tools.

We also conducted a small evaluation of Tell Me More's difference metrics module. Preliminary user studies (Iacobelli, Birnbaum, and Hammond 2010) and the results reported here suggest that Tell Me More is able to find genuinely new information that is also perceived as relevant by users. Future work will target recall, the ranking of new information, and a more comprehensive system evaluation.

Acknowledgements

This research is based upon work supported by the National Science Foundation under Grant No. III-0917261.

References

Budzik, J.; Hammond, K. J.; and Birnbaum, L. 2001. Information access in context. *Knowledge-Based Systems* 14(1-2):37–53.

Downey, D.; Broadhead, M.; and Etzioni, O. 2007. Locating complex named entities in web text. In *In Proc. of IJCAI*.

Etzioni, O.; Cafarella, M.; Downey, D.; Popescu, A.-M.; Shaked, T.; Soderland, S.; Weld, D. S.; and Yates, A. 2005. Unsupervised named-entity extraction from the web: an experimental study. *Artif. Intell.* 165(1):91–134.

Fredkin, E. 1960. Trie memory. *Commun. ACM* 3(9):490–499.

Gabrilovich, E.; Dumais, S.; and Horvitz, E. 2004. Newsjunkie: providing personalized newsfeeds via analysis of information novelty. In *WWW '04: Proceedings of the 13th international conference on World Wide Web*, 482–490. New York, NY, USA: ACM Press.

Iacobelli, F.; Birnbaum, L.; and Hammond, K. J. 2010. Tell me more, not just "more of the same". In *IUI '10: Proceeding of the 14th international conference on Intelligent user interfaces*, 81–90. New York, NY, USA: ACM.

Käki, M. 2005. Findex: search result categories help users when document ranking fails. In *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, 131–140. New York, NY, USA: ACM.

Keppel, G., and Underwood, B. 1962. Proactive inhibition in short-term retention of single items. *Journal of Verbal Learning and Verbal Behavior* 1(3):153–161.

Lafferty, J.; McCallum, A.; and Pereira, F. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proc. 18th International Conf. on Machine Learning*, 282–289. Morgan Kaufmann, San Francisco, CA.

Li, W., and McCallum, A. 2005. Semi-supervised sequence modeling with syntactic topic models. In *AAAI-05, The Twentieth National Conference on Artificial Intelligence*.

Milne, D., and Witten, I. H. 2008. Learning to link with wikipedia. In *CIKM '08: Proceeding of the 17th ACM conference on Information and knowledge management*, 509–518. New York, NY, USA: ACM.

Park, S.; Lee, S.; and Song, J. 2010. Aspect-level news browsing: Understanding news events from multiple viewpoints. In *Intelligent User Interfaces (IUI2010)*, 41–50.

Salton, G., and Buckley, C. 1988. Term-weighting approaches in automatic text retrieval. In *Information Processing and Management*, 513–523.

Schiffman, B., and Mckeown, K. R. 2004. Columbia university in the novelty track at trec 2004. In *Proceedings of the TREC 2004*.

Schiffman, B. 2005. *Learning to identify new information*. Ph.D. Dissertation, Columbia University.

Soboroff, I., and Harman, D. 2003. Overview of the TREC 2003 novelty track. In *Proceedings of TREC-2003*. Citeseer.

Sweeney, S.; Crestani, F.; and Losada, D. 2008. 'show me more': Incremental length summarisation using novelty detection. *Information Processing & Management* 44(2):663–686.

Wagner, E. J.; Liu, J.; Birnbaum, L.; and Forbus, K. D. 2009. Rich interfaces for reading news on the web. In *IUI '09: Proceedings of the 13th international conference on Intelligent user interfaces*, 27–36. New York, NY, USA: ACM.

Yerva, S. R.; Miklós, Z.; and Aberer, K. 2010. Towards better entity resolution techniques for web document collections. In *Proceedings of 1st International Workshop on Data Engineering meets the Semantic Web, co-located with ICDE 2010*.