Filtering Information Extraction via User-Contributed Knowledge

Thomas Lin, Oren Etzioni, James Fogarty
Computer Science & Engineering
University of Washington
Seattle, WA 98195, USA
{tlin, etzioni, jfogarty}@cs.washington.edu

Abstract

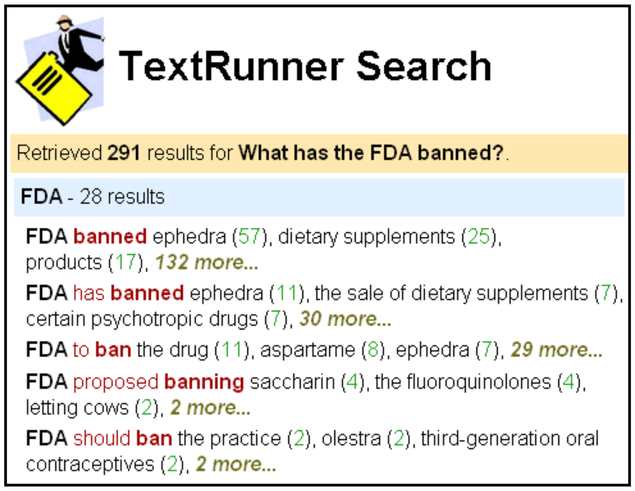
Large repositories of knowledge can enable more powerful AI systems. Information Extraction (IE) is one approach to building knowledge repositories by extracting knowledge from text. Open IE systems like TextRunner [Banko et al., 2007] are able to extract hundreds of millions of assertions from Web text. However, because of imperfections in extraction technology and the noisy nature of Web text, IE systems return a mix of both useful, informative facts (e.g., "*the FDA banned ephedra*") and less informative statements (e.g., "*the FDA banned products*").

This paper investigates using user-contributed knowledge from Wikipedia and from TextRunner website visitors to train classifiers that automatically filter extracted assertions. In a study of human ratings of the interestingness of TextRunner assertions, we show that our approach substantially enhances the quality of results. Our relevance feedback filter raises the fraction of *interesting* results in the top thirty from 41.6% to 64.1%.

1 Introduction

Information extraction (IE) is a subfield of natural language processing that seeks to obtain structured information from unstructured text. IE can be used to automate the tedious and error prone process of collecting facts from the Web. Open IE is a relation-independent form of IE that scales well to large corpuses. Figure 1 presents the output of the TextRunner Open IE system [Banko et al., 2007] in response to the question "*What has the FDA banned?*". TextRunner homes in on such answers as "*ephedra*" and "*most silicone implants*" and frees people from sifting through many Web pages to find the desired answers.

Unfortunately, extraction engines, like search engines, intermix relevant information with irrelevant information. This problem is exacerbated in IE systems because they use heuristic methods to extract phrases that are meant to denote entities and relationships. Thus, in response to the above question, an extraction engine like TextRunner also returns such uninformative answers as "*products*" and "*the drug*". Experiments presented in this paper show that people find 58.4% of the thirty top-ranked answers returned by TextRunner to be uninformative.



Retrieved **291** results for **What has the FDA banned?**.

FDA - 28 results

FDA banned ephedra (57), dietary supplements (25), products (17), *132 more...*

FDA has banned ephedra (11), the sale of dietary supplements (7), certain psychotropic drugs (7), *30 more...*

FDA to ban the drug (11), aspartame (8), ephedra (7), *29 more...*

FDA proposed banning saccharin (4), the fluoroquinolones (4), letting cows (2), *2 more...*

FDA should ban the practice (2), olestra (2), third-generation oral contraceptives (2), *2 more...*

Figure 1. TextRunner results for the question "*What has the FDA banned?*". This paper examines the filtering of such results to focus on *interesting* assertions.

Extraction engines therefore could be improved by filtering based on models of which extracted assertions are of interest and which are not. See Figure 2 for an overview of this idea. Of course, the notion of *interestingness* is subjective, personal, and context specific. Nevertheless, any system that returns ranked results, from Google to TextRunner, either implicitly or explicitly utilizes a model of *what is interesting* in its ranking function.

A challenge here is that while people can identify what is interesting to them, it is less clear how computers can do this algorithmically. We could implement several theories of interestingness from psychology such as *complexity*, *novelty*, *uncertainty*, and *conflict* [Silvia, 2006], but it is unclear which, if any, is best. The most accurate method would be if we could have people go through and specify which of the extracted assertions are of interest.

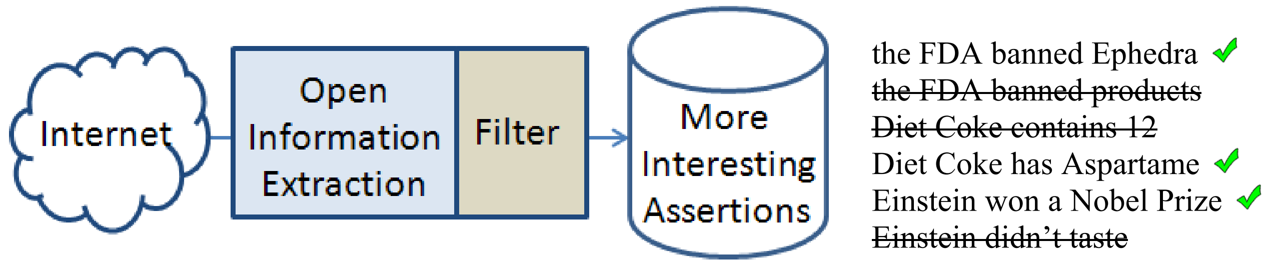

Internet
Open Information Extraction
Filter
More Interesting Assertions
the FDA banned Ephedra ✓
~~the FDA banned products~~
~~Diet Coke contains 12~~
Diet Coke has Aspartame ✓
Einstein won a Nobel Prize ✓
~~Einstein didn't taste~~

Figure 2. Filtering the output of Open IE enables it to better focus on extracted assertions that are more *interesting*.

However, as TextRunner has over 800 million extracted assertions, it would be prohibitively expensive to hand-label every assertion. Instead, we collect labels on a small but general subset of the data, and then generalize by using this as training data for a classifier filter that combines basic features and psychology theories. We also leverage large available repositories of user-contributed knowledge. Using Wikipedia Infobox matching to automatically classify assertions, we are able to inexpensively generate additional sizable sets of assertions that are likely to be interesting and not interesting to train a classifier. Figure 3 shows our process for using user-contributed knowledge.

This paper proposes several models of *what is interesting*, presents our implementation of these models as filters, and reports on measurements of their efficacy on a sample of queries. The main contributions of this paper are to:

- Introduce several practical models of interestingness that, when implemented as filters, offer substantial improvements over the existing technique of sorting assertions by frequency. These models are informed by previous work, theories, and user-contributed knowledge. Our models could be easily adapted to aid other Web-based extraction systems, such as PowerSet.
- Utilize a machine-learning method that leverages all the models, together with relevance feedback, to filter out uninteresting assertions resulting from extraction.
- Report on the first study of interestingness in extraction. For this study we compare the efficacy of our models to each other and the TextRunner baseline. Among other findings, we show that our filtering significantly improves the fraction of interesting results contained within TextRunner's top thirty results from 41.6% interesting to 64.1% interesting.

The remainder of this paper is organized as follows. Section 2 introduces TextRunner and related work. Section 3 describes our models of interestingness and how we operationalize them as filters. Section 4 presents a study evaluating those models. We then conclude with a discussion.

2 Background

The TextRunner system crawls the Web and extracts information as triples that take the form **(entity, relation, entity).** The *relation* string is meant to denote the relationship between the two *entities*. For example, if the sentence *"Franz Kafka was born in Prague, at the time part of Austria"* were found on a webpage, then one extraction would be (*"Franz Kafka", "was born in", "Prague"*).

This extraction process is based in an automatically trained extractor [Banko and Etzioni, 2008]. The extractor is domain independent and relation independent (Open IE), and has been run on 500 million high-quality webpages yielding over 800 million extractions. These are indexed in Lucene and can be queried by entity or relationship.

As shown in Figure 1, TextRunner results are returned ranked by frequency. TextRunner ranks results by frequency because, all other things being equal, extractions that appear frequently on high-quality Web pages are more likely to be correct [Downey et al., 2005]. However, experience has shown that this technique also yields many vague or otherwise uninteresting assertions.

2.1 Related Work

Traditional Information Extraction

A key aspect of this study is that in order to scale better to the full Web, we are studying models that can improve the interestingness of Web extractions in a domain independent and relation independent way. This is important because lexical rules (e.g. "all assertions about what companies Microsoft has bought are interesting") might work well for particular domains or relations but not apply more generally.

In traditional IE systems, system developers pre-specify relations of interest and then provide training examples. For example, the NAGA system [Kasneci et al., 2008] has considered methods for evaluating quality of web extractions, but their work is grounded in a graph representation based on the specific set of relationships that they chose to extract. This limited set of relationships meant that they could only evaluate 12 of 50 queries for one of their benchmarks.

Systems that Consider Interestingness

The general concept of using interestingness as a metric has value and applicability to a wide range of domains. For instance, Flickr recently launched a new feature[1] for identifying "Interestingness" in photos on its site. The Flickr

[1] http://www.flickr.com/explore/interesting/

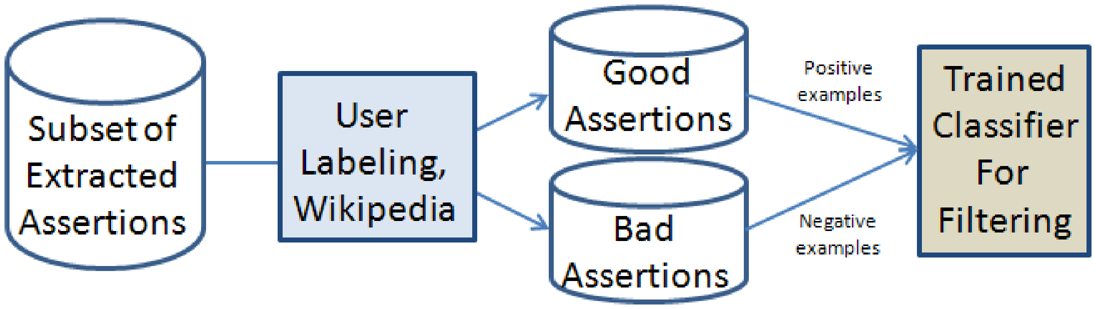


Figure 3. We train our filter using examples of good and bad assertions from TextRunner. We have filters based on both user labels and a Wikipedia Infobox approach that allowed us to easily leverage a large amount of user-contributed knowledge.

notion is based on social feedback such as click data and comments, supporting the idea that people care about what's interesting and leave indirect clues to where interesting content can be found. We use a similar concept later in learning from how people populate Wikipedia infoboxes.

Similarly, automated mathematical discovery programs require a notion of interestingness in order to identify which potential conjectures and concepts will be of interest to people. Colton and Bundy's survey [1999] identified several key concepts that these programs tended to use in deciding what would be interesting, including plausibility, novelty, surprisingness, comprehensibility and complexity. Liu et al. [2000] found that unexpected database association rules are more interesting to users.

What Makes Text Interesting?

Beyond the psychological work mentioned in the introduction, there has also been research into what makes text more interesting. It is important that text be the right level of complexity. Sentences with concrete words were found to be more interesting than abstract sentences [Sadoski et al., 1993]. Texts that are more coherent and easier to comprehend are more interesting [Schraw, 1997]. Prior knowledge in the subject generally increases interest. These ideas inform some of our classifier features later.

3 What's Interesting?

This section describes the problem of *interestingness*, then introduces three practical models for identifying interesting assertions. For each model, we first present an intuition behind characteristics that can make assertions interesting. We then operationalize those characteristics so that we can express them algorithmically.

We want to capture the interesting assertions, but what exactly does this mean? At the most general level, we define interesting assertions to be those that a person may find useful or engaging. For any particular query (e.g., "*Einstein*"), the extent to which possible assertions are interesting may vary greatly. A good set of results might, for example, include a mix of biographical facts like "*Einstein was born in Germany*" and other interesting facts like "*Einstein's favorite color was blue*". On the other hand, "*Einstein turned 15*" or "*Einstein wrote the paper*", would be less interesting because they express little useful information.

In a discussion of what is interesting, personalization is one approach to consider. Different people will find different topics to be interesting. We consider personalized notions of interesting to be a future direction, but currently focus on what characteristics make an assertion broadly interesting to a variety of people.

3.1 Specific Assertions

One quality of interesting assertions is that they tend to provide more *specific* information. For example, "*Albert Einstein taught at Princeton University*" is more interesting than "*Albert Einstein taught at a university*" because identifying Princeton as the university is informative. We hypothesize this is one characteristic that can make assertions interesting more broadly in TextRunner.

To operationalize this quality, we define a *specific* assertion as an assertion that either relates multiple proper nouns or an assertion that contains a year. If an assertion relates multiple proper nouns, it is specific because it expresses information about one specific entity relative to another. Similarly, an assertion that contains a year is specific because it contains specific temporal information.

3.2 Distinguishing Assertions

Another quality of interesting assertions might be providing *distinguishing* information about an object. This is related to novelty and surprisingness. Einstein may be a physicist who was born in Germany, but what really sets him apart and makes him interesting are his contributions to relativity theory and that he won the Nobel Prize. Conversely, assertions that do not set an object apart from other objects are often uninteresting.

We operationalize this notion of *distinguishing* using a technique similar to TF-IDF (term frequency – inverse document frequency) weighting [Salton and Buckley, 1988]. In IR, term frequency refers to the number of times a term occurs in a document. For our term frequency component, we define *AssertionFrequency* as the number of times an assertion occurs in the TextRunner set of assertions (e.g., the number of times TextRunner found that "*Einstein won the Nobel Prize*"). For our document frequency component, we define *ObjectFrequency* as the number of times the object (e.g., "*the Nobel Prize*") appears in a sample of ten million random TextRunner assertions. We define an *AFOFRatio(Extraction)* as follows[2]:

$$AFOFRatio(E) = \frac{AssertionFrequency(E)}{ObjectFrequency\big(object(E)\big) + 1}$$

For assertions, the *AFOFRatio* compares how often the assertion appears with how often we would expect the assertion to appear given its object. If the object has extremely high *ObjectFrequency* (e.g., a common word like "*food*"), the *AFOFRatio* will be low. If the object has extremely low *ObjectFrequency* (e.g., a misspelling or obscure term), then the *AFOFRatio* will be high. In the case of average *ObjectFrequency*, the *AFOFRatio* will reflect whether the assertion appears more often than one would normally expect.

Informal experimentation confirmed that extremely low *AFOFRatio* values often indicate an assertion is too vague to be interesting, while the very highest *AFOFRatio* values generally indicated assertions that were not well formed or well expressed. We chose a middle range ($1 < AFOFRatio \leq 10$) that seemed to generally yield interesting assertions from the *distinguishing* perspective.

3.3 Basic Assertions using Wikipedia

The final quality of interesting assertions that we focus on here are *basic* facts, definitional assertions that, for example, might be interesting to a person learning about an object. A person learning about Einstein, for example, might look up such facts as "*Einstein was a physicist*" or "*Einstein was born in 1879*". Although it would be difficult to define all-encompassing rules for what makes an assertion *basic*, we can take advantage of the fact that user-contributed knowledge bases like Wikipedia provide high quality examples of *basic* knowledge. Many Wikipedia articles contain infoboxes, tabular summaries of *basic* information about objects. Our operationalization of *basic* assertions, is therefore based in learning a classifier to identify assertions like those that human editors have decided to include in infoboxes.

Training a classifier with Wikipedia infobox data allows us to automatically leverage enough existing high quality human-generated knowledge to bootstrap learning [Wu and Weld, 2007]. We first obtain training data by automatically finding TextRunner assertions that reflect the information found in infoboxes. Notable people generally have populated Wikipedia infoboxes, so we obtain our training data using queries for famous people. We used the DBPedia Wikipedia Infobox database [Auer et al., 2007] and applied a series of filters to isolate a set of notable people with high quality infoboxes. We then matched the infobox data against TextRunner to obtain 1,584 assertions that reflected information found in infoboxes, to use as positive training examples, and an even larger set of assertions that did not match infoboxes, to use as negative training examples.

We train our *basic* classifier using around ten domain-independent features, such as the number of words in the assertion, whether the assertion relates proper nouns, and the estimated frequency of the assertion's object argument in TextRunner. Lexical features (those specific to the query terms, such as learning that any assertion with the relation "was born in" is interesting), are intentionally omitted because we are interested in a *generally applicable* classifier that is effective regardless of whether it was trained on assertions similar to those that it will classify (e.g., "was born in" is useless on assertions about fruits). An experiment showed that the lexical features enable greater precision at the cost of reduced recall and generality.

If we already have Wikipedia and we hypothesize that the infobox attributes are what's interesting, then why not just use all the Wikipedia data instead of using Web extraction? The key point here is that compared to the full Web, Wikipedia is incomplete. Many entities do not have Wikipedia articles, either because they have not been written yet or because they do not belong in a general encyclopedia. Even when an entity does have a Wikipedia article, often the infobox is incomplete or even missing. Also, while infobox attributes are good starting point for *basic* knowledge, there exist many additional similar attributes that also express *basic* knowledge. Web extraction has the potential for much greater coverage, both in terms of entities covered and attributes per entity.

4 Evaluating Human Ratings of Interesting

In order to evaluate our *specific*, *basic*, and *distinguishing* models, we used them each as the basis for filters that discard TextRunner results that fail to satisfy each model. To assess the quality of each filter, we conducted a study to collect human ratings of the interestingness of assertions.

4.1 Method and Procedure

We first selected a set of ten study query terms including famous people (*Albert Einstein, Bill Gates, Thomas Edison*), other proper nouns (*Beijing, Brazil, Microsoft, Diet Coke*), improper nouns (*sea lions*), and also relationship queries (*invented, destroyed*). This query set is meant to provide a varied sample of the sorts of queries for which TextRunner can provide interesting results. Our analyses are based on the top thirty assertions resulting from each of these queries, because the top results have the greatest impact on utility and about thirty results can be seen at a glance on a TextRunner results page.

As a baseline for comparison, we first obtain the number of times each assertion was found by TextRunner, and so

[2] We add 1 in the denominator to prevent possible division by 0.

our *AssertionFrequency* condition examines the thirty most frequently occurring assertions. We next obtain assertions for our *specific*, *distinguishing*, and *basic* conditions by applying each of our filters in order of assertion frequency, discarding results that fail the filter, until we obtain thirty results that satisfy the filter. The study therefore focuses on 1200 assertions (10 queries * 4 conditions * 30 assertions).

We recruited 12 study participants (7 female), who had a variety of backgrounds including math, marketing, finance, music, and nursing. Participants were each asked to rate 200 assertions on a scale from 1 (labeled "Least Interesting") to 5 (labeled "Most Interesting"). Assertions were presented one at a time, drawn randomly without replacement between participants. We gathered two or three ratings for every assertion, helping to account for individual differences in what people consider interesting.

4.2 Results

We analyze participant ratings of interestingness using a mixed-model analysis of variance. We model our variable of interest, *condition* (values *AssertionFrequency*, *simple*, *distinguishing*, and *basic*), as a fixed effect. To account for learning or fatigue effects, we model *trial number* as a fixed effect. Similarly, we account for the possibility that how long a person viewed an assertion might impact their rating by modeling *time to rate* as a fixed effect. Finally, we account for variations in the interestingness of queries and variations in the ratings given by different people by modeling both *query* and *participant* as random effects.

We found no significant effect of either *trial number* or *time to rate*, and so remove both of them from the remainder of our analysis. The omnibus test reveals a significant main effect of *condition* ($F(4, 3542) = 15.6, p < .0001$), leading us to investigate pairwise differences. We use Tukey's Honestly Significant Difference (HSD) procedure to account for increased Type I error in unplanned comparisons. This shows *basic* yielded the most interesting assertions, significantly more interesting than *AssertionFrequency* ($F(1,3545) = 55.0, p < .0001$), *specific* ($F(4,3539) = 7.7, p \approx .005$), and *distinguishing* ($F(1,3547) = 10.3, p \approx .001$). Our other filters also significantly improved interestingness, as both *specific* ($F(1,3544) = 21.2, p < .0001$) and *distinguishing* ($F(1,3539) = 18.7, p < .0001$) were significantly more interesting than *AssertionFrequency*.

4.3 Relevance Feedback

Although our results showed that *basic* assertions are the most interesting and that all of our filters yield results that are significantly more interesting than *AssertionFrequency*, inspection of our data suggested that our filters identify *different* interesting assertions. We found that only 21% of the *interesting* assertions would be identified by all three filters. We therefore consider whether a learning-based method, using a classifier to combine information from all three filters, might perform better than any single filter.

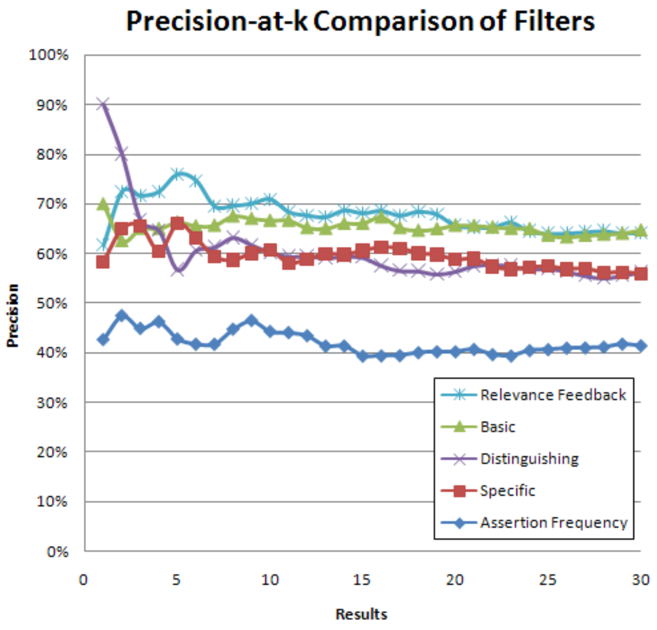


Figure 4. Our trained filters led to significantly higher mean average precisions for whether top assertions were interesting. Relevance Feedback (67.9%) was the best (p ≈ .005). *Basic* (65.4%) was second best (p < .0001). *Specific* (59.5%) and *distinguishing* (60.3%) were also better (p < .0001) than Assertion Frequency (the results without any filtering), which had the lowest mean average precision at 41.9%.

In this case our training data is from study participants, but the same ideas would apply to collecting interactive feedback from users via a Web interface. We have developed such an interactive Web interface for TextRunner where people browsing query results can click on assertions to highlight them and specify whether they are of interest.

In order to simplify this and our remaining analyses, we first reduce our five-point scale to a binary classification. We define ratings of *4* or *5* to be *interesting*, define ratings of *1* or *2* to be *not interesting*, and ignore ratings of *3*. This discretization creates a nearly even split of our collected human labels, which we then use as positive and negative training examples for a relevance feedback classifier.

We make the output of our *specific*, *distinguishing*, and *basic* filters available as features to this classifier. We also provide the same features that are used by the *basic* classifier. Because we are interested in a *generally applicable* classifier of *interesting* assertions, we evaluate trained relevance feedback classifiers using ten-fold cross-validation such that we only test on the assertions from query terms not used to train the filter. This allows us to estimate performance on queries for which the system has not been trained, as we would expect improved performance on any queries for which the system has been trained.

Several classifiers from the WEKA toolkit [Witten and Frank, 2005] all had comparable *precision at k* values, averaging over ten-fold cross-validation, from k=1 to k=30. Decision Tree [Quinlan, 1993] was slightly better than the rest with an average precision at 67.9%. The *precision at k* measure illustrates the percentage of the first *k* results that are *interesting*, and is an appropriate and important measure because it corresponds to the quality of the top results.

4.4 Analysis

Figure 4 plots the *precision at k* for the *relevance feedback* decision tree classifier versus our *specific*, *distinguishing*, and *basic* filters as well as against *AssertionFrequency*. To test for difference between these curves, we conduct an analysis of variance for the precision at each plotted point, treating *condition* and *k* as fixed effects. The omnibus test reveals a significant main effect of *condition* ($F(4, 4) = 285$, $p < .0001$), leading us to investigate pairwise differences. We use Tukey's HSD procedure to account for increased Type I error in unplanned comparisons. This shows that *relevance feedback* yields significantly more interesting assertions than *specific* ($F(1,144) = 95.4$, $p < .0001$), *distinguishing* ($F(1,144) = 78.6$, $p < .0001$), *basic* ($F(1,144) = 8$, $p \approx .005$) and *AssertionFrequency* ($F(1,144)=926$, $p<.0001$).

The largest differences in Figure 4 are between our filter-based approaches and TextRunner's original use of *AssertionFrequency*, indicating the advantage of filtering. The classifier filters trained with user-contributed knowledge (*relevance feedback* and *basic*) performed significantly better than all other approaches, indicating the utility of user-contributed knowledge for this task. Our *relevance feedback* classifier achieves a precision at 30 of 64.1% and a mean average precision of 67.9%. This is comparable to human level performance, as we measured inter-annotator agreement in our label set to be approximately 70%.

5 Conclusions

Extraction engines such as TextRunner are a promising avenue towards improving Web search and generating large knowledge bases. However, such systems are currently hamstrung by the fact that they often return uninformative results that are vague or uninteresting. Web extraction systems are particularly prone to this problem because of the general methods they use to extract entities and relationships [Banko and Etzioni, 2008]. This paper has developed filters based on user contributions that allow TextRunner to better focus on assertions that are *interesting*. These filters raised the average percentage of interesting results on a sample of queries from 41.6% to 64.1%.

Leveraging user-contributed knowledge to improve the quality of Open IE presents an interesting synergy. Open IE draws knowledge from the Web, and at the same time it contributes back to the Web in terms of smarter searching and question answering abilities and the ability to empower better AI applications via a large knowledge base. Our filtering uses knowledge from Wikipedia to enable higher quality Open IE, which in turn could lead to good contributions back to Wikipedia via projects such as those that use IE on Wikipedia article text to populate Wikipedia Infoboxes [Hoffmann et al., 2009].

One avenue of future work is to incorporate research on entity ranking [Zaragoza et al., 2007], which might provide valuable additional input in areas such as entity generality.

Acknowledgements

This research was supported in part by NSF grants IIS-0803481 and IIS-0812590, ONR grant N00014-08-1-0431, and Google and was carried out at the University of Washington's Turing Center.

References

[Auer et al., 2007] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, Z. Ives. DBPedia: A Nucleus for a Web of Open Data. In *Proc ISWC 2007*.

[Banko and Etzioni, 2008] M. Banko and O. Etzioni. The Tradeoffs Between Open and Traditional Relation Extraction. In *Proc ACL 2008*.

[Banko et al., 2007] M. Banko, M.J. Cafarella, S. Soderland, M. Broadhead, O. Etzioni. Open Information Extraction from the Web. In *Proc IJCAI 2007*.

[Colton and Bundy, 1999] S. Colton and A. Bundy. On the Notion of Interestingness in Automated Mathematical Discovery. In *Proc AISB 1999*.

[Downey et al., 2005] D. Downey, O. Etzioni, S. Soderland, A Probabilistic Model of Redundancy in Information Extraction. In *Proc IJCAI 2005*.

[Hoffmann et al., 2009] R. Hoffmann, S. Amershi, K. Patel, F. Wu, J. Fogarty, D. Weld, Amplifying Community Content Creation Using Mixed-Initiative Information Extraction. In *Proc CHI 2009*.

[Kasneci et al., 2008] G. Kasneci, F. Suchanek, G. Ifrim, M. Ramanath, G. Weikum. NAGA: Searching and Ranking Knowledge. In *Proc ICDE 2008*.

[Liu et al., 2000] B. Liu, W. Hsu, S. Chen, Y. Ma, Analyzing the Subjective Interestingness of Association Rules. *IEEE Intelligent Systems 15*, 47-55, 2000.

[Quinlan, 1993] J. R. Quinlan, *C4.5: Programs for Machine Learning*. Morgan Kaufman, 1993.

[Sadoski et al., 1993] M. Sadoski, E.T. Goetz, J.B. Fritz. A causal model of sentence recall: Effects of familiarity, concreteness, comprehensibility and interestingness. *Journal of Reading Behavior, 25*, 5-16, 1993.

[Salton and Buckley, 1988] G. Salton and C. Buckley. Term-weighting approaches in automatic text retrieval, *Information Processing & Management, 24*, 1988.

[Schraw, 1997] G. Schraw. Situational interest in literary text. *Contemporary Educational Psychology, 22*, 436-456, 1997.

[Silvia, 2006] P.J. Silvia, *Exploring the Psychology of Interest*. Oxford University Press, New York, NY, 2006.

[Witten and Frank, 2005] I.H. Witten and E. Frank. *Data Mining: Practical machine learning tools and techniques*, 2nd Edition, Morgan Kaufmann, 2005.

[Wu and Weld, 2007] F. Wu, and D. Weld. Autonomously Semantifying Wikipedia. In *Proc CIKM 2007*.

[Zaragoza et al, 2007] H. Zaragoza, H. Rode, P. Mika, J. Atserias, M. Ciaramita, G. Attardi. Ranking Very Many Typed Entities on Wikipedia. In *Proc CIKM 2007*.