Fun Facts: Automatic Trivia Fact Extraction from Wikipedia

David Tsurel
The Hebrew University of Jerusalem
dmtsurel@cs.huji.ac.il

Dan Pelleg
Yahoo Research, Israel
pellegd@acm.org

Ido Guy
Yahoo Research, Israel
Ben-Gurion University of the Negev
idoguy@acm.org

Dafna Shahaf
The Hebrew University of Jerusalem
dshahaf@cs.huji.ac.il

ABSTRACT

A significant portion of web search queries directly refers to named entities. Search engines explore various ways to improve the user experience for such queries. We suggest augmenting search results with *trivia facts* about the searched entity. Trivia is widely played throughout the world, and was shown to increase users' engagement and retention.

Most random facts are not suitable for the trivia section. There is skill (and art) to curating good trivia. In this paper, we formalize a notion of *trivia-worthiness* and propose an algorithm that automatically mines trivia facts from Wikipedia. We take advantage of Wikipedia's category structure, and rank an entity's categories by their trivia-quality. Our algorithm is capable of finding interesting facts, such as Obama's Grammy or Elvis' stint as a tank gunner. In user studies, our algorithm captures the intuitive notion of "good trivia" 45% higher than prior work. Search-page tests show a 22% decrease in bounce rates and a 12% increase in dwell time, proving our facts hold users' attention.

1. INTRODUCTION

> Libraries may be full of facts, but finding beautiful trivia in those dry, dusty stacks is like panning for gold. The glittering grains are few and far between. As the introduction to one early trivia book says, there is a difference between "the flower of trivia and the weed of minutiae." Or, to put it another way, all trivia may be facts, but not all facts are capital-T Trivia. I can't spell out the difference, but I know it's there. "Comedian Albert Brooks attended Carnegie Tech in Pittsburgh" is a fact. So is "Comedian Albert Brooks is five-foot-ten-inches tall" – not that interesting unless you're his tailor. But "Comedian Albert Brooks had to change his name because he was born Albert Einstein"? Ah. That's trivia.
>
> Ken Jennings [1]

WSDM 2017, February 06-10, 2017, Cambridge, United Kingdom

DOI: http://dx.doi.org/10.1145/3018661.3018709

Today, anybody with a smartphone has access to far more information than even "Jeopardy!" champion Ken Jennings could ever recall. Despite this, trivia games are more popular than ever. From board games to mobile apps, newspapers and pubs, trivia is widely played throughout the world.

In this paper, we tackle the problem of **automatically extracting trivia facts** from Wikipedia. This task, while seemingly lighthearted, has real-world applications. In particular, we are motivated by its application to *search*.

In many cases, search is a goal-driven activity: first, there is some information need, specific or general (e.g., the current time in Alofi, or alternatively, a cure for Zika). We then approach a seemingly-omniscient mechanism, in the form of a search engine, to answer the need. We wait, get the answer, and then go on — presumably towards some other well-defined information need. This is the prevailing view in the information-retrieval community, which places the search-engine and the user at opposing roles: one is the producer of search topics, while the other is the consumer, and acts merely as a passive librarian, looking up the facts. However, many users do not share this utilitarian view [2]. For them (or for some of their queries), search is an exploratory activity, and at some stages of the information-gathering process, less-than-relevant results are welcome [3].

With this in mind, we note that a significant portion — over 50% — of web search queries directly refers to named entities [4, 5]. Modern web search engines explore various technologies to improve user experience for these types of queries. Such technologies include clustering of search results for disambiguation, related entity recommendation, and the presentation of rich "entity cards", which include key aspects of the entity, directly on the search engine results page (SERP) [6]. Recently, Miliaraki et al. [7] demonstrated a search system which, in addition to the search results, surfaced entities related — but not necessarily directly — to the query. Importantly, surfacing other entities was proven to be an effective vehicle for drawing searchers to an exploratory activity, thereby increasing engagement.

We propose augmenting search results with **trivia facts** that are related to the searched entity. We believe trivia facts could contribute to the user experience around entity searches; even a small impact on this type of queries can translate into a significantly improved user experience.

There are multiple reasons to believe that trivia can indeed contribute to the user experience. Business case studies [8] have shown that trivia helps increase user engagement

and revenue. A man who tweets random facts has over 18 million followers, and makes about $500,000 a year from sponsored links [9].

However, there is skill (and art) to coming up with good trivia. In a recent experiment, professional trivia curators managed to find trivia facts for merely ten entities per day, on average [10]. The process is expensive and hard to scale.

In order to automate the process of finding good trivia, one needs to characterize the notion of what is *trivia-worthy*. In this work, we introduce and formalize two criteria that characterize good trivia: *surprise* and *cohesiveness*. Using our formulation, we propose an algorithm that automatically extracts trivia facts from Wikipedia articles. We take advantage of the category structure of Wikipedia, and rank an entity's categories by their trivia-quality. Our algorithm is capable of finding interesting facts, such as Obama's Grammy Award win, or Elvis' stint as a tank gunner.

User studies with crowd-sourced workers show that our algorithm produces facts that capture the intuitive notion of "good trivia". In another study, we bought ads on search pages and measured the level of interest in trivia in a real life scenario. Trivia facts were generally found to arouse interest, while better trivia facts attract page views with lower bounce rates and longer dwell time.

2. PROBLEM FORMULATION

Our goal is to automatically find trivia facts about entities. We first consider possible sources of such facts. Wikipedia is a natural choice for this purpose because of its wide coverage. However, Wikipedia articles are written in natural language; working with short textual units (e.g., sentences), one must deal with anaphora resolution, long-range references, and other context-related problems. Thus, we focus on Wikipedia's *category structure*. Categories are sets of articles with a shared topic, such as "History of France", "Philosophy of mind", or "Biological concepts". An article can belong to multiple categories. For example, Barack Obama's categories include "Presidents of the United States", "Columbia University alumni", and "Grammy Award winners" (see Figure 1). Importantly for us, categories are cleaner than sentences, while often capturing the most interesting aspects of the article [11, 12].

Given an article, we want our algorithm to rank its categories, such that the top-ranked categories should be most suitable for the trivia section. Therefore, we need to formalize the notion of *trivia-worthy*. The Merriam-Webster dictionary defines trivia as *"Unimportant facts or details. Facts about people, events, etc., that are not well-known."*

One possible direction for detecting a trivia-worthy category would be to choose the category with the smallest number of articles. Presumably, a small category indicates a rare and unique property of an entity, and would be an interesting trivia fact. However, testing this path has shown it focuses on properties that were too narrow, in several senses: Most often, the smallest category focuses on a very specific identity aspect, usually obscure and uninteresting - "Muhammad Ali is an alumni of Central High School in Louisville, Kentucky" is not a good trivia fact - the specific high school has no importance to the reader and does not reflect on Ali's character. In other cases, when the entity belonged to a well-known family, band or group, the smallest category captured a well-known aspect of the entity, for example "Michael Jackson was a member of the The Jackson 5".


Categories: Barack Obama | Obama family | 1961 births | 20th-century American writers | 20th-century scholars | 21st-century American politicians | 21st-century American writers | 21st-century scholars | African-American academics | African-American Christians | African-American lawyers | African-American non-fiction writers | African-American politicians | African-American United States presidential candidates | African-American United States Senators | American book publishers (people) | American civil rights lawyers | American community activists | American expatriates in Indonesia | American gun control advocates | American legal scholars | American male writers | American memoirists | American Nobel laureates | American people of English descent | American people of Irish descent | American people of Kenyan descent | American people of Luo descent | American political writers | American Protestants | Columbia University alumni | Critics of Islamophobia | Democratic Party Presidents of the United States | Democratic Party (United States) presidential nominees | Democratic Party United States Senators | Grammy Award winners | Harvard Law School alumni | Illinois Democrats | Illinois lawyers | Illinois State Senators | International opponents of apartheid in South Africa | Irish diaspora politicians | Living people | Nobel Peace Prize laureates | Occidental College alumni | Politicians from Chicago, Illinois | Politicians from Honolulu, Hawaii | Politicians of Luo descent | Presidents of the United States | Progressivism in the United States | Punahou School alumni | Recipients of the Presidential Medal of Distinction of Israel | United States presidential candidates, 2008 | United States presidential candidates, 2012 | United States Senators from Illinois | University of Chicago Law School faculty | Writers from Chicago, Illinois

Figure 1: Categories from Obama's Wikipedia page

Indeed, trivia facts are often centered around uncommon knowledge. In other words, trivia facts are *surprising*. For example, everybody who knows Obama probably knows he belongs to the "Presidents of the United States" category, but many people would be surprised to learn that he won a Grammy. On the other hand, we want facts that are also interesting and not obscure.

We begin by formulating our first property - *surprise*.

2.1 Surprise

Surprise measures how unusual it is for a given article to belong to a category. In other words, we would need to define a similarity metric between an article a and a category C. Since a category is a collection of articles, our main building block will be a similarity metric between articles. We denote article-article similarity by $\sigma(a, a')$, and defer its exact implementation details to Section 3.

Next, we extend the similarity between articles to similarity between articles and *categories*. A category C is a set of articles. We define the similarity of an article a to category C as the average similarity between a and the articles of C:

$$\sigma(a, C) = \frac{1}{|C| - 1} \sum_{a \neq a' \in C} \sigma(a, a')$$

An article is surprising w.r.t. a category if its average similarity to the other articles is low. Thus, we define surprise as the inverse of the average similarity:

$$surp(a, C) = \frac{1}{\sigma(a, C)}$$

For example, consider 1940s Hollywood film actress Hedy Lamarr. When ranking her Wikipedia page categories by surprise, the top 5 results (out of 18) are, in order (top is most surprising):

"20th-century Austrian people"
"Women in technology"
"Radio pioneers"
"American anti-fascists"
"American people of Hungarian-Jewish descent"

2.2 Cohesiveness

Looking at Hedy Lamarr's most surprising categories, some of them do not fit our intuitive idea of trivia. For example, regarding her Austro-Hungarian descent: Wikipedia chose to list it because of its charter to record the minutiae of famous people's biographies, but the detail itself is not particularly trivia-worthy.

However, other aspects of her life are less mundane. For example, this Hollywood star had also invented radio encryption (more precisely, she had patents in frequency-hopping, spread-spectrum technology[1]). Yet, our notion of surprise ranks the fact similarly to how it ranks her (less exciting) Austrian pedigree.

Note that all five categories are, indeed, surprising; many people who know Hedy Lamarr as a movie actress are probably not aware of these aspects of her life. Therefore, we conclude that surprise is not enough.

However, the thing that makes the group "Radio pioneers" more suitable for our purposes is somewhat harder to define. Intuitively, being of Hungarian-Jewish descent seems more arbitrary than being a radio pioneer; being Austrian says less about the person than being a woman in technology. We believe that elusive notion is related to the *cohesiveness* of the group: People born in Austria can come from all walks of life. Look at the Wikipedia category "20th-century Austrian people", and you will find lawyers, architects, painters, physicians, and World War I diplomats. On the other hand, radio pioneers seem like a more close-knit group.

Thus, we define our second metric, *cohesiveness*. We define cohesiveness of category C as the average similarity between pairs of articles from C:

$$cohesive(C) = \frac{1}{\binom{|C|}{2}} \sum_{a \neq a'} \sigma(a, a')$$

When ranking Hedy Lamarr's categories by cohesiveness, the top-5 categories are, in order:

"Metro-Goldwyn-Mayer contract players"
"Actresses from Vienna"
"Austrian film actresses"
"20th-century Austrian actresses"
"American film actresses"

These categories are indeed cohesive, in the sense that they are not arbitrary details. However, to be fair, this list is not yet a good set of trivia facts. For that, we have to consider both surprise and cohesiveness, which we do below.

2.3 Tying it Together

We have just defined two properties – surprise and cohesiveness. We now wish to combine them into a notion of *trivia-worthiness*.

Figure 2 shows the categories of Hedy Lamarr. The x-axis represents cohesiveness, and the y-axis represents surprise. Intuitively, a category is trivia-worthy if it is high on both surprise and cohesiveness scores. Therefore, we would like to define a score that is monotonic in both dimensions.

We define the trivia-worthiness of a category C w.r.t. article a as the product of cohesiveness and surprise.

$$trivia(a, C) = cohesive(C) \cdot surp(a, C)$$

[1]As well as inventions in traffic lights and soft drinks. Did you know *that*?

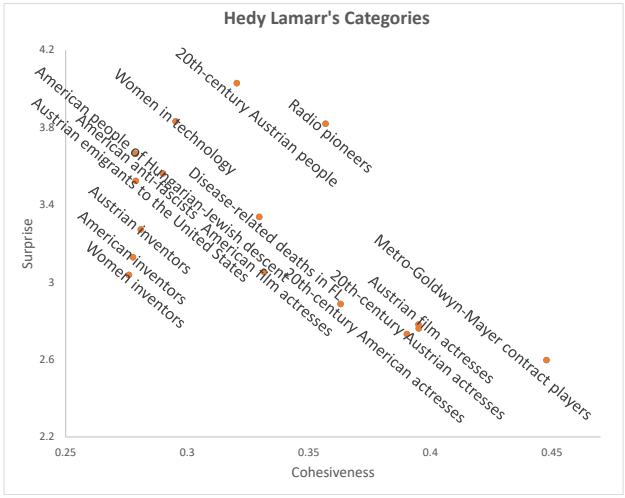

Figure 2: Cohesiveness plotted against surprise for the categories of Hedy Lamarr. To reduce label clutter, we omit some of the labels.

There are many other ways to combine the two scores. However, multiplication has a natural interpretation. Note that surprise is defined as the inverse of the average similarity of a to the category. Thus, the multiplicative formula measures how similar the article is to the category, compared to the average similarity of articles from the same category. In other words, we measure whether the article is more similar or less similar to the category than was expected.

$$trivia(a, C) = \frac{cohesive(C)}{\sigma(a, C)}$$

A value of *trivia* around one means, by definition, that the average distance between a and C is similar to the cohesiveness of C, which indicates that the article is typical for that category: it is similar to other articles in the category just as much as the average article.

A value of *trivia* much lower than one indicates that the article is more similar to other articles than the average, and could be an *exemplar*. It is a prominent member of the category, and would not be good trivia.

Now, a value of *trivia* higher than one means that the article in question is not so similar to the category. In some sense, it is an *outsider*, which might make good trivia.

Example. Figure 3 illustrates our ideas. The figure shows similarity among articles from the "Democratic Party Presidents of the United States" category (left), compared to articles from "Grammy Award winners" (right). Each node is an article, and the edge weight represents similarity. Obama is in the center of both graphs.

First, we look at the edges from Obama to other articles. Being a Grammy winner is much more *surprising*, as demonstrated by the thinner edges between Obama's node and the rest of the graph — a similarity score of only 0.241, compared to 0.601 for the democratic presidents.

Next, we move on to *cohesiveness*. The cohesiveness of the two different categories can be seen in Figure 3 as the average thickness of the edges. The Grammy winners have a cohesiveness score of 0.398, as they are mostly well known

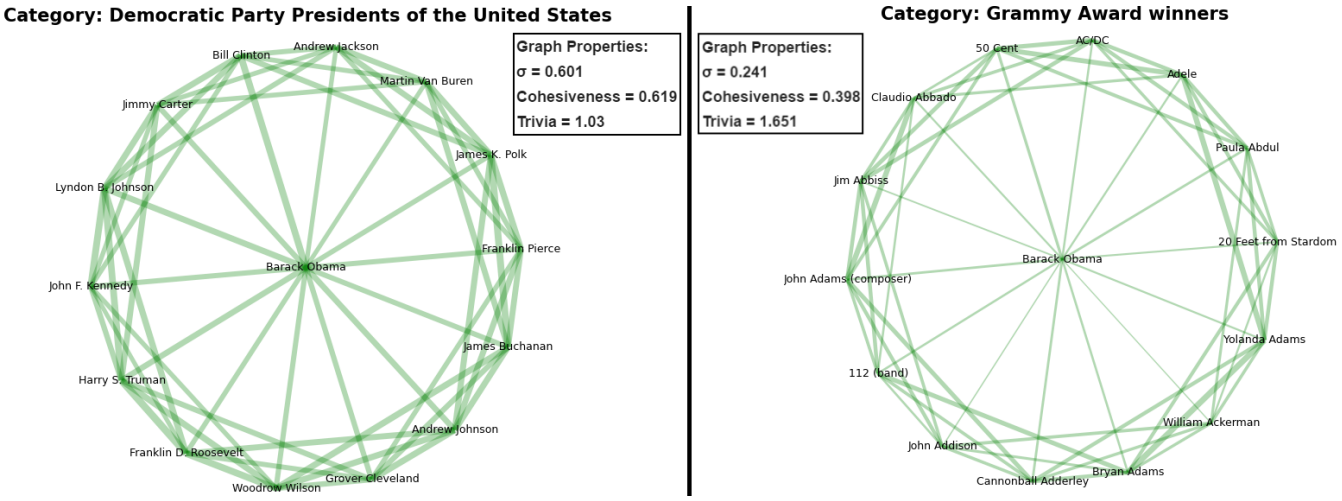

Figure 3: Similarity graphs for two categories containing Barack Obama. Thicker edges are more similar. For visualization reasons, not all nodes and edges are shown.

musicians, although from different genres. Presidents, on the other hand, are all leading US politicians, and are a well-knit group, earning them a score of 0.619.

Finally, we look at Obama's edges, compared to the rest of the edges in the category. We can see that in the democratic presidents graph, Obama has high similarity to the other nodes, but so is the average similarity in the graph. The *trivia* value is $\frac{cohesive(C)}{\sigma(a,C)} = \frac{0.619}{0.601} = 1.030$, which indicates that Obama is a typical article in the category. In the Grammy category, however, the similarity between Obama and the other nodes is much weaker than the average similarity in the category. The *trivia* value is $\frac{0.398}{0.241} = 1.651$, which indicates that Obama is not a typical Grammy winner.

3. ALGORITHM

For a formal description of our method, see Algorithm 1. K is a parameter, described below. In each category, we compute cohesiveness and surprise with regard to the input article, and combine them into a trivia score.

An important component in our formulation was an article-article similarity metric, $\sigma(a, a')$ (computed by the function ArticleSimilarity). We now discuss its implementation details.

3.1 Article Similarity

When choosing a metric for article comparison, standard methods such as cosine similarity between term frequency vectors proved to be inadequate. For example, when comparing the articles "Apple" and "Orange", cosine similarity was only 0.026, even though both are fruit. "Apple" and "Barack Obama" had a higher similarity, 0.059. There are two main problems underlying the usual similarity methods:

- We are looking for relatively **broad similarity** (for example, "both people sing in rock bands"); we do not necessarily need details to be similar. Long articles, in particular, can add a significant amount of noise.
- Term frequency vectors look for exact matches between terms, so many **semantic similarities** are lost (even after stemming and normalization).

Algorithm 1 Top Trivia algorithm

function TopTrivia(*inputArticle*)
  for every category C of *inputArticle* **do**
    $surprise \leftarrow$ Surprise(*inputArticle*, C)
    $cohesiveness \leftarrow$ Cohesiveness(C)
    $C.trivia \leftarrow cohesiveness * surprise$
  return category C with maximum *trivia* score

function Surprise(*inputArticle*, *category*)
  $sum, count \leftarrow 0$
  for every article $a \neq inputArticle$ in category C **do**
    $similarity \leftarrow$ ArticleSimilarity(*inputArticle*, a)
    $sum \leftarrow sum + similarity$
    $count \leftarrow count + 1$
  $similarityToCategory \leftarrow sum/count$
  $surprise \leftarrow 1/similarityToCategory$
  return *surprise*

function Cohesiveness(*category*)
  $sum, count \leftarrow 0$
  for every pair of articles $a1 \neq a2$ in category C **do**
    $similarity \leftarrow$ ArticleSimilarity($a1, a2$)
    $sum \leftarrow sum + similarity$
    $count \leftarrow count + 1$
  $cohesiveness \leftarrow sum/count$
  return *cohesiveness*

function ArticleSimilarity(article1, article2)
  $K \leftarrow 10$
  $T1 \leftarrow$ TopTFIDF($article1, K$)
  $T2 \leftarrow$ TopTFIDF($article2, K$)
  $similarity \leftarrow \sigma(article1, article2)$ using equation 3.1
  return *similarity*

To address the first problem (**broad similarity**), we do not use all words in an article. Instead, we compute TF-IDF scores for all words in the documents. TF-IDF measures how important a word is in a document, given a corpus. For our

Table 1: Top TF-IDF Terms

Sherlock Holmes	Dr. Watson	Hercule Poirot
holmes	watson	murder
sherlock	holmes	christie
watson	sherlock	hastings
adventures	adventures	detective
detective	doyle	novels
conan	portrayed	curtain
doyle	conan	belgian
stories	stories	solve
scarlet	detective	mystery
bohemia	doctor	adaptation

corpus, we used a sample of 10,000 articles from the English Wikipedia. We used standard text normalization techniques such as stemming, stopword removal and case folding. We removed terms appearing in less than 10 documents.

We restrict ourselves to the top K TF-IDF terms of each article. In our experiments we used $K = 10$, after testing showed it balanced between noise reduction and retaining important information. For example, table 1 displays the top 10 TF-IDF terms for the articles "Sherlock Holmes" and "Dr. Watson", the main characters in the detective stories of Arthur Conan Doyle, and the article "Hercule Poirot", another fictional detective from stories by Agatha Christie.

While these words seem to capture the gist of the three characters, there are almost no exact matches. Holmes and Poirot, for example, share exactly one word – detective. There are, however, many semantically related words, such as murder/detective, novels/adventures.

To address the problem of **semantic similarity**, we compute word similarity $\sigma(w_1, w_2)$ using the word2vec representation [13]. We used a pre-trained model, trained on a Google News corpus of about 100 billion tokens using a neural network to produce a 300-dimensional vector space of word embeddings. Word similarity is in the range [-1,1], where higher values indicate stronger similarity. Interestingly, word2vec is known to capture semantic similarities [14, 15]. For example:

$$\sigma(\text{“}christie\text{”}, \text{“}doyle\text{”}) = 0.529$$
$$\sigma(\text{“}novels\text{”}, \text{“}adventures\text{”}) = 0.423$$
$$\sigma(\text{“}curtain\text{”}, \text{“}scarlet\text{”}) = 0.163$$

To compute similarity between articles, we look at the similarities of their top words. Let T_1 and T_2 be the sets of top-K TF-IDF terms for two articles, a_1 and a_2. For each TF-IDF term in T_1, we find the most similar term in T_2 (and vice versa, to keep the definition symmetric) and sum up these similarities. We use a weighted formula to give more weight to terms with higher TF-IDF scores and normalize the result to the range [-1,1]:

$$\sigma(a_1, a_2) = \tag{3.1}$$
$$\frac{1}{\mathcal{Z}} \sum_{i=1}^{K} w(i) \cdot \left(\max_{1 \le j \le K} \sigma(T_1[i], T_2[j]) + \max_{1 \le j \le K} \sigma(T_2[i], T_1[j]) \right)$$

We experimented with several weighting schemes, and chose a linear one: $w(i) = K - i + 1$, with normalization factor $\mathcal{Z} = 2 \cdot \binom{K+1}{2}$.

When comparing articles using this method, the articles "Apple" and "Orange" had a similarity score of 0.3, compared to only 0.11 for "Apple" and "Barack Obama". "Sherlock Holmes" and "Hercule Poirot" had a similarity score of 0.513.

3.2 Practical Considerations

To improve efficiency and allow reuse of intermediate results, we used *caching* throughout our algorithm, writing values to both memory and file system. To increase speed, we *parallelized* the algorithm so it could (1) process several articles simultaneously, and (2) rate the trivia-worthiness of several categories simultaneously for each article.

When computing surprise and cohesiveness for large categories, one needs to compute similarity between $O(n^2)$ pairs of articles. To speed the computation up, we randomly sample a subset of the articles instead, and computed similarity between all pairs in the subset. In our experiments, we found that 50 articles are usually enough to obtain results that are very close to the results of using the full set of articles.

4. EVALUATION

In this section, we evaluate our algorithm empirically. We have compared the following algorithms:

- **Wikipedia Trivia Miner (WTM)** [10]: A ranking algorithm over Wikipedia sentences, which learns the notion of interestingness using domain-independent linguistic and entity based features. The supervised ranking model is trained on existing user-generated trivia data available on the Web.
- **Top Trivia**: The highest ranking category in our algorithm ranking.
- **Middle-ranked Trivia**: Using middle-of-the-pack ranked categories, as ranked by our algorithm.
- **Bottom Trivia**: Using the lowest-ranked categories by our algorithm.

We collected article and category data for our experiments via the Wikipedia web API using the Pywikibot framework [16] and an adapted version of the Wiki2Plain interface [17].

We created a dataset of 400 popular Wikipedia articles about people, based on a list of the most viewed pages over the week of July 10-16, 2016 [18]. The list contains a diverse range of popular people, including politicians, sportspeople, scientists, actors, writers, singers, historical figures and other people of interest.

However, the popularity of a page does not necessarily indicate that it contains good trivia. To ensure a fair comparison, we restricted ourselves to pages where both our algorithm and WTM found good trivia. In particular, we selected articles for which the trivia fact had a score in the top 50% of facts in both our algorithm and the WTM rankings. This resulted in a dataset of trivia facts for 109 articles.

For every article, we produced the single trivia fact for each of the algorithms. The textual format for our facts is "a is in the group C". Table 2 shows an example of the trivia facts produced for the article "Barack Obama". Our data and code are available at https://github.com/DMTsurel/FunFacts

4.1 Trivia Evaluation Study

Evaluation of trivia facts is a subjective matter. Therefore, a key part of our evaluation is based on a user study we performed using crowd-sourced work.

For each of the 109 articles, we computed four trivia facts: our top, middle and bottom facts, as well as WTM. Each fact was presented to five crowd workers, for a total of 2180

Table 2: Top fact returned by each algorithm for the article "Barack Obama"

Algorithm	Fact
Top	Barack Obama is in the group of Grammy Award winners
Middle	Barack Obama is in the group of African-American lawyers
Bottom	Barack Obama is in the group of Obama family
WTM	Besides his native English, Obama speaks some basic Indonesian, having learned the language during his four childhood years in Jakarta.

evaluations. To increase reliability, we restricted workers' location to the US, and their approval rate to above 80%. We made our task available to *Mechanical Turk Masters* only (a qualification given by Mechanical Turk to workers who perform consistently well across a wide range of tasks).

The workers were presented with the fact and asked to express their level of agreement with the following statements:

- Trivia-worthiness: *"This is a good trivia fact".*
- Surprise: *"This fact is surprising".*
- Personal knowledge: *"I knew this fact before reading it here".*

Workers could agree or disagree with each statement, or reply that they could not understand the fact. For each statement, the *majority opinion* of a fact is the answer agreed on by at least 50% of workers. Five evaluations had missing answers for the trivia-worthiness statement. In two of these, no majority was reached because of the missing answer, so they were removed from the results.

Results. Figure 4 shows the percentage of facts that a majority of users ranked as trivia-worthy, by algorithm. As expected, the facts ranked "top" by our algorithm outperform the "middle", which outperform "bottom". Our algorithm proved to be significantly better at finding trivia-worthy facts than the WTM baseline: 56%, compared to only 38.5% for WTM ($p < 0.01$, Pearson's chi-squared test).

When looking at the type of majority (3, 4 or 5 users), we notice that 32.8% of the facts ranked as trivia-worthy by our algorithm achieved a perfect agreement (5 users), compared to only 11.9% for WTM.

Facts that could not be understood are the worst type of facts – not only are users presented with facts that are not good trivia, but they leave the users confused. Our algorithm had no such facts, compared to 23.9% for the WTM baseline and 8.3% for our bottom baseline. This is an advantage of using categories for facts, as they are self-contained pieces of information. The WTM baseline used sentences from the Wikipedia text, which are sometimes left out of context even after trying to remove such sentences using co-reference resolution [10]. For example, the WTM baseline fact for Leonardo DiCaprio was "The project achieved a worldwide box office take of $147 million." Users did not know what project was referenced in this sentence, so they could not understand the fact. Our bottom baseline also had several confusing facts. For example, the "William Shakespeare" category was ranked worst for the article William Shakespeare, and users were confused by the fact "William Shakespeare is in the group of William Shakespeare".

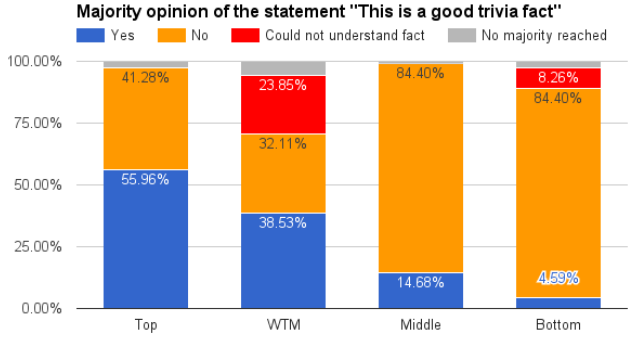

Figure 4: Majority opinion about facts being trivia-worthy, by algorithm

We examined instances where our algorithm failed while WTM managed to find an interesting fact. Our algorithm examines only facts formulated as categories, so it will miss anecdotes that do not pertain to a set of articles. For example, most workers did not think the fact "Beyonce is in the group of Shoe designers", found by our algorithm, was trivia worthy. The trivia fact suggested by WTM was ranked as good trivia: "On January 7, 2012, Beyonce gave birth to her first child, a daughter, Blue Ivy Carter, at Lenox Hill Hospital in New York". The latter fact is too specific to be captured by a category.

Results for surprise were similar (Figure 5). Facts ranked as the top category were more surprising to users than those in the middle and bottom. 50.5% of our algorithm's top results were surprising to users, 47.7% were not surprising, and there were 0% where users could not understand the fact. The WTM algorithm had 39.5% of its facts ranked as surprising, 32.1% as not surprising, and 23.8% could not be understood ($p < 0.01$, Pearson's chi-squared test).

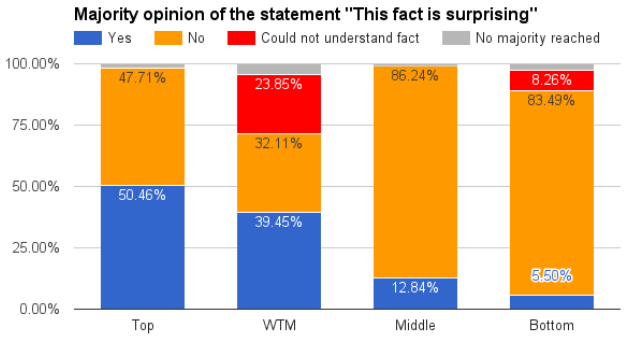

Figure 5: Majority opinion about facts being surprising, by algorithm

Figure 6 shows the percentage of facts that a majority of users knew previously. Almost all facts in both our algorithm's top choice and WTM were previously unknown to users. However, when choosing the middle or bottom categories, the likelihood of being familiar with the facts is much higher. This indicates that our ranking method works well in terms of filtering out well-known facts.

To test our hypothesis that good trivia is based on the element of surprise, we consider the contingency table of trivia-worthiness and surprise (Table 3). We see that there is in-

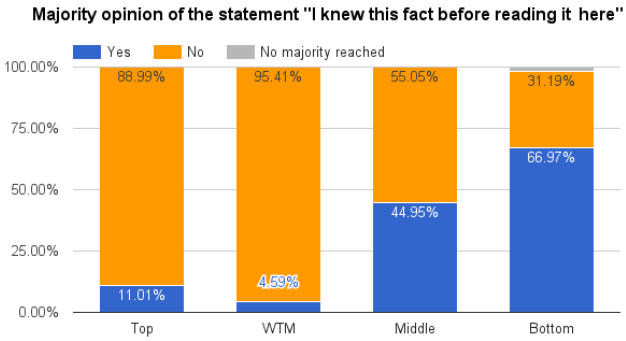

Figure 6: Majority opinion about personal knowledge of facts, by algorithm

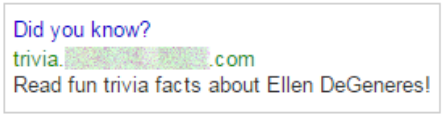

Figure 7: An example ad used in the engagement study.

deed strong correlation between surprise and trivia-worthiness (One-Tailed Fisher Exact Probability Test, $p < 10^{-50}$).

Table 3: Trivia-worthiness and surprise

	Surprising	Not Surprising
Trivia-worthy	102	22
Not Trivia-worthy	12	250

4.2 Engagement Study

In addition to the direct approach used in the first study, we conducted an additional study to indirectly measure how engaging trivia facts are.

In this study, we were targeting users who searched the entities in our dataset on the Web. We used Google AdWords [19] to buy ads pertaining to these entities (see Figure 7). When users clicked an ad, they were directed to one of three variations of a landing page. The variations corresponded to the **Top Trivia**, **Bottom Trivia** and **WTM** algorithms. Note that the ad itself was the same for all three conditions. Furthermore, we turned off Google's optimization algorithms, to ensure that the users would be uniformly distributed between the conditions.

Each page began with the trivia fact extracted by the corresponding algorithm, and then provided a mirror of the original Wikipedia article, with the fact highlighted (Figure 8). Users were directed to paragraphs in the article text that contained broader context. For category-based facts, these paragraphs were chosen as those with the highest σ value, compared to the category title. "Click here for another random fact" allowed users to navigate to other trivia pages generated by the same algorithm. We conjectured that better trivia facts will engage the users more.

Results. We collected nearly 500 clicks throughout the experiment. Key measures for ad success are click-through rate and bounce rate [20]. We also measured average dwell time of users on the site.

CTR is the percentage of users clicking on an ad. We use

Figure 8: A partial screenshot of a landing page. It is a mirror of the Wikipedia page, with the trivia fact at the beginning (top) and the corresponding parts of the article highlighted (bottom).

CTR to gauge users' level of interest in trivia in a real-life search scenario. In our study, CTR was inconsistent over different days, but overall averaged to 0.8%. Baseline CTR values in search ads is considered commercially sensitive information, so it is difficult to find non-normalized reference points. According to a recent analysis [21], this value does indicate willingness of users to explore trivia facts.

Next, we compared our three groups of users. For the Bottom Trivia condition, 52% of users bounced immediately out of the site (under 5 seconds). WTM had 47%, and Top Trivia 37%. Some of the bounce rate might be explained by misguided clicks. For example, an Ellen DeGeneres ad was shown to people whose search included "Ellen", and was clicked on by people who searched for other Ellens.

Average time on the site for the users who did not bounce was 30.7 seconds for Bottom Trivia, 43.1 seconds for WTM and 48.5 seconds for Top Trivia. We used the one-sided Mann-Whitney U test to test whether that difference was significant. Our hypothesis was that Top Trivia users stayed longer on the site. Top Trivia was indeed better than Bottom Trivia ($p \approx 0.02$). However, the difference from WTM was not statistically significant ($p \approx 0.15$).

We note that dwell time is a coarse measure for trivia quality, and there are other reasons that could explain a longer dwell time. For example, WTM facts were often long sentences ("Mandel has mysophobia (a pathological fear of contamination/germs) to the point that he does not shake hands with anyone, including enthusiastic contestants on Deal or No Deal, unless he is wearing latex gloves"). Other times, the WTM facts were somewhat cryptic (Andy Kaufman's fact was "Keep that in mind when you call"), as can also be seen by the number of people who could not understand them in the Mechanical Turk experiment (Figure 4). Both those reasons might prompt people to spend more time on the page – either processing longer sentences, or scrolling down to understand the context of obscure sentences.

5. DISCUSSION AND FUTURE WORK

In the following section, we discuss our algorithm's limitations and potential extensions.

Limitations. We note that the proposed algorithm works well for human entities. However, there are domains where categories do not include many interesting trivia facts, such as movies or cities. (For example, consider the London categories: London, British capitals, Capitals in Europe, Populated places established in the 1st century, Port cities and towns in England, Staple ports)

In these domains, our algorithm's ability to find good trivia facts is limited. Note that even in such domains, false positives can generally be avoided by a *trivia* score threshold, as the categories in these cases are homogeneous and have *trivia* values close to 1.

We note that our algorithm currently generates only a single-template mold ("X is a member of group Y") that does not appeal to users. A possible direction towards breaking the template would be to return from categories to natural language sentences: given the title of a category C we attempt to find in the article text a sentence S that contains similar information, using a variant of the σ function. Testing this function on the "Grammy Award winners" category for Barack Obama gave the following sentence as the top result: "Obama won Best Spoken Word Album Grammy Awards for abridged audiobook versions of Dreams from My Father in February 2006 and for The Audacity of Hope in February 2008."

A related problem is that of turning trivia facts into trivia *questions*. For example, "Barack Obama won a Grammy award" is a good trivia fact, but turning it into a question is not straightforward. "Who won a Grammy award?" or "What did Barack Obama win?" are not good trivia questions, as they both have too many valid answers. One way to generate good questions would be to contrast a well-known category with the trivia-worthy category: "Which US president is a Grammy award winner?"

Other Applications. Our goal in this paper was to find the best trivia fact for a given article. However, our algorithm can be useful for other tasks as well. For example, the top category for Abraham Lincoln was "American Vegetarians". This is indeed surprising, but turns out to be historically false [22]. Detection of anomalous information could be useful in removing inaccurate claims from Wikipedia, thereby increasing its reliability.

In addition, the proposed *surprise* metric can also be used to detect the most surprising article for a given category – or the least surprising one. For example, our algorithm detected that the least surprising article in the "British television chefs" category was Gordon Ramsay, who is indeed very prominent in that category.

We largely framed the trivia insertion problem as one that piggybacks on top of search. However, search today is just another function the smartphone performs. The boundaries between entertainment and information are blurred (as evidenced by the increase of music video queries in voice search [23]). Combined with location data to help identify if the user is ready for listless exploration, the technology presented here could help build proactive educational agents.

Extensions. There are multiple dimensions we can add to our formulation, most important of which is probably *personalization*. Bob Marley's Syrian-Jewish descent might be more interesting to people who are Syrian, Jewish (or both). A growing body of work looks into personalization in recommender systems. Mejova et al. [24] suggest personalized trivia facts as a method of breaking the "Filter Bubble" of social networks and increasing user interest in geographically remote countries. Young people and older people might enjoy different facts: in [7], there is a strong match between the age of the suggested person entity and the age of the searcher. The diversity of countries and cultures can create unique perspectives on what is obvious and what is surprising [25].

In the absence of data about personal preferences, we can use *popularity* as an aggregated signal. The number of page views can indicate general level of interest in a category. Temporal popularity patterns can be used to bias our algorithm (e.g., showing somebody's Irish descent just before Saint Patrick's Day).

6. RELATED WORK

There is relatively little work in Computer Science focusing on trivia. In the work closest to ours, Prakash et al. [10] introduced the WTM algorithm. This work used supervised learning to extract linguistic and entity-based features from a labeled dataset derived from the IMDb (Internet Movie Database) trivia section. Unlike our method, WTM algorithm does not utilize Wikipedia's structure. In addition, its application is limited to domains where large free labeled databases such as IMDb exist. WTM is used as a baseline in Section 4. Despite being simpler, our algorithm finds better trivia facts.

Merzbacher [26] tackled a related problem of mining trivia questions from a database. The questions are constructed by composing together functions (for example, the standard relational algebra operators). Serban et al. [27] applied a neural network architecture on the Freebase knowledge base to transduce template-based relations into natural-language questions. In contrast to our approach, these methods assume a relational database structure, and thus have limited applicability.

There is a large body of work devoted to the more general questions of surprise, interestingness and anomaly detection [28]. For example, Byrne and Hunter [29] develop a logic-based framework that translates structured news reports into formulas, identifying as interesting those that violate consistency or contradict axiomatic beliefs and expectations. Gamon et al. [25] consider the concept of interestingness as a user's desire to know more about a topic. By observing web-browsing logs of transitions between Wikipedia articles they construct a probabilistic model that learns latent semantic features that are interesting to users. McGarry [30] conducts a literature survey of interestingness measures used in knowledge discovery, divided into objective statistical measures and subjective measures based on user beliefs or a specific domain. Malone et al. [31] define differential ratio rules to detect interesting patterns in spatio-temporal data. The technique uses ratios of features over time to detect change, similar to our definition of *trivia-worthiness*.

In recent years, the importance of *serendipity* as a measure for the success of recommender systems has grown, as one of the most prominent "beyond-accuracy" measures [32, 33]. McNee et al. [32] define it as the experience of getting an unexpected and fortuitous item. Desrosiers and Karypis [34] tie it with helping users find something inter-

esting they might not have otherwise discovered. Herlocker et al. [35] define serendipity as the extent to which the items are both attractive and surprising to users. Sun et al. [36] define serendipity in social networks context as messages unexpected from the sender and relevant to the receiver. Producing serendipitous recommendations is performed by various means, such as promoting items that have both a strong positive and a strong negative prediction scores [37] or items that are well connected, in a graph representation, both to the user's preferred items and to unrelated items [38]. Evaluating serendipity is also a challenge. A recent study of social-stream item recommendation [39] directly asked participants if they found the recommended items surprising in order to asses serendipity. We ask a similar question about trivia facts in the user study conduced as part of our own evaluation.

Serendipity was also explored in the context of *search*. Recently, Miliarki et al. [7] demonstrated a search module which explores entities related to a search query. It was proven to be an effective vehicle for drawing searchers to an exploratory activity. Interestingly, the highest engagement was registered when the mentioned entity was a person (as compared to location, or a movie). This serves as further motivation for our suggestion of augmenting entity search results with related trivia facts. An issue left open is how to predict the response in advance – that is, whether the user is "focused" or "exploratory".

7. CONCLUSIONS

The prevailing view in the information-retrieval community sees the search engine as a passive librarian, looking up the facts. However, many users today expect the search engine to provide not just information, but also entertainment. We believe that with the advent of new search interfaces, it is time to re-examine the idea of adding serendipity to search.

Building on the popularity of entities (and in particular person entities) in current search, we propose an algorithm to identify facts about people as *trivia-worthy*. Specifically, we examine group membership in Wikipedia categories and rank them according to two dimensions: *surprise* and *cohesiveness*. Surprise relates to our prior on the person belonging to a given group, while cohesiveness ensures that said group is indeed interesting to begin with. We present a simple algorithm that is capable of discovering interesting facts, such as Hedy Lamarr's inventions.

We performed two kinds of user studies. First, directly and with crowd-sourced work, we show that our facts are judged as good trivia, surprising and previously unknown. Compared to prior work, our facts are judged as 27% more surprising, and 45% better trivia facts. Second, by buying ads on search pages, we show that our trivia facts attract page views with longer dwell times (12%) and lower bounce rates (22%), as compared to the baseline.

This application, while seemingly lighthearted, can lead to higher engagement of users searching for named entities. If successful, even a small impact on this type of queries can translate into a substantial improvement in user experience, and possibly transfer to other domains of human activity, like education.

8. REFERENCES

[1] Ken Jennings. *Brainiac: adventures in the curious, competitive, compulsive world of trivia buffs*. Villard Books, 2007.

[2] Paul André, Jaime Teevan, and Susan T. Dumais. From x-rays to Silly Putty via Uranus: Serendipity and its role in web search. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pages 2033–2036, New York, NY, USA, 2009. ACM.

[3] Amanda Spink, Howard Greisdorf, and Judy Bateman. From highly relevant to not relevant: examining different regions of relevance. *Information Processing & Management*, 34(5):599–621, 1998.

[4] Peter Mika. Entity search on the web. In *Proc. WWW Companion*, pages 1231–1232, 2013.

[5] Xiaoxin Yin and Sarthak Shah. Building taxonomy of web search intents for name entity queries. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 1001–1010, New York, NY, USA, 2010. ACM.

[6] Horatiu Bota, Ke Zhou, and Joemon M. Jose. Playing your cards right: The effect of entity cards on search behaviour and workload. In *Proc. CHIIR*, pages 131–140, 2016.

[7] Iris Miliaraki, Roi Blanco, and Mounia Lalmas. From "Selena Gomez" to "Marlon Brando": Understanding explorative entity search. In *24th International World Wide Web Conference (WWW 2015)*, Florence, Italy, May 2015.

[8] Using trivia and quiz products to engage your customer, http://www.slideshare.net/woverstreet/using-trivia-and-quiz-products-to-engage-your-customer. [Online; accessed 17-July-2016].

[9] This 25-year-old makes $500,000 a year tweeting random facts, http://www.cnbc.com/2016/07/16/25-year-old-kris-sanchez-makes-500000-a-year-from-uberfacts.html. [Online; accessed 17-July-2016].

[10] Abhay Prakash, Manoj K. Chinnakotla, Dhaval Patel, and Puneet Garg. Did you know?: Mining interesting trivia for entities from wikipedia. In *Proceedings of the 24th International Conference on Artificial Intelligence*, IJCAI'15, pages 3164–3170. AAAI Press, 2015.

[11] Sergey Chernov, Tereza Iofciu, Wolfgang Nejdl, and Xuan Zhou. Extracting semantics relationships between Wikipedia categories. *SemWiki*, 206, 2006.

[12] Vivi Nastase and Michael Strube. Decoding Wikipedia categories for knowledge acquisition. In *AAAI*, volume 8, pages 1219–1224, 2008.

[13] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781*, 2013.

[14] Marco Baroni, Georgiana Dinu, and Germán Kruszewski. Don't count, predict! a systematic comparison of context-counting vs. context-predicting semantic vectors. In *Proceedings of Association for Computational Linguistics (ACL)*, volume 1, 2014.

[15] Tom Kenter and Maarten de Rijke. Short text similarity with word embeddings. In *Proceedings of the 24th ACM International on Conference on*

Information and Knowledge Management, CIKM '15, pages 1411–1420, New York, NY, USA, 2015. ACM.

[16] MediaWiki. Manual:Pywikibot — Mediawiki, The Free Wiki Engine, https://www.mediawiki.org/w/index.php?title=Manual:Pywikibot&oldid=2176177, [Online; accessed 17-July-2016].

[17] joksnet. Wiki2Plain, http://stackoverflow.com/a/4461624. [Online; accessed 17-July-2016].

[18] Wikipedia. User:West.andrew.g/Popular pages — Wikipedia, The Free Encyclopedia, https://en.wikipedia.org/w/index.php?title=User:West.andrew.g/Popular_pages&oldid=730185650. [Online; accessed 17-July-2016].

[19] Andrew E Goodman. *Winning Results with Google AdWords*. McGraw-Hill/Osborne, 2005.

[20] D Sculley, Robert G Malkin, Sugato Basu, and Roberto J Bayardo. Predicting bounce rates in sponsored search advertisements. In *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1325–1334. ACM, 2009.

[21] Display advertising clickthrough rates, http://www.smartinsights.com/internet-advertising/internet-advertising-analytics/display-advertising-clickthrough-rates/. [Online; accessed 17-July-2016].

[22] Mike Hudak. Abraham Lincoln: vegetarian and animal rights advocate? - a review of the evidence. *Broome County History Bulletin (Fall 2009, vol. 8, no. 2)*, 2009.

[23] Ido Guy. Searching by talking: Analysis of voice queries on mobile web search. In *Proceedings of the 39th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '16, pages 35–44, New York, NY, USA, 2016. ACM.

[24] Yelena Mejova, Javier Borge-Holthoefer, and Ingmar Weber. Bridges into the unknown: Personalizing connections to little-known countries. In *Proceedings of the 33rd Annual ACM Conference on Human Factors in Computing Systems*, CHI '15, pages 2633–2642, New York, NY, USA, 2015. ACM.

[25] Michael Gamon, Arjun Mukherjee, and Patrick Pantel. Predicting interesting things in text. In *COLING 2014, 25th International Conference on Computational Linguistics, Proceedings of the Conference: Technical Papers, August 23-29, 2014, Dublin, Ireland*, pages 1477–1488, 2014.

[26] Matthew Merzbacher. Automatic generation of trivia questions. In *International Symposium on Methodologies for Intelligent Systems*, pages 123–130. Springer, 2002.

[27] Iulian Vlad Serban, Alberto García-Durán, Çaglar Gülçehre, Sungjin Ahn, Sarath Chandar, Aaron C. Courville, and Yoshua Bengio. Generating factoid questions with recurrent neural networks: The 30m factoid question-answer corpus. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*. The Association for Computer Linguistics, 2016.

[28] Varun Chandola, Arindam Banerjee, and Vipin Kumar. Anomaly detection: A survey. *ACM computing surveys (CSUR)*, 41(3):15, 2009.

[29] Emma Byrne and Anthony Hunter. Man bites dog: looking for interesting inconsistencies in structured news reports. *Data & Knowledge Engineering*, 48(3):265 – 295, 2004.

[30] Ken McGarry. A survey of interestingness measures for knowledge discovery. *Knowl. Eng. Rev.*, 20(1):39–61, March 2005.

[31] James Malone, Kenneth McGarry, and Chris Bowerman. Performing trend analysis on spatio-temporal proteomics data using differential ratio data mining. In *Proceedings of the 6th EPSRC Conference on Postgraduate Research in Electronics, Photonics, Communications and Software (PREP 2004)*, pages 103–105, 2004.

[32] Sean M. McNee, John Riedl, and Joseph A. Konstan. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In *Proc. CHI EA*, pages 1097–1101, 2006.

[33] Mouzhi Ge, Carla Delgado-Battenfeld, and Dietmar Jannach. Beyond accuracy: Evaluating recommender systems by coverage and serendipity. In *Proc. RecSys*, pages 257–260, 2010.

[34] Christian Desrosiers and George Karypis. A comprehensive survey of neighborhood-based recommendation methods. In *Recommender Systems Handbook*, pages 107–144. Springer, 2011.

[35] Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1):5–53, January 2004.

[36] Tao Sun, Ming Zhang, and Qiaozhu Mei. Unexpected relevance: An empirical study of serendipity in retweets. In Emre Kiciman, Nicole B. Ellison, Bernie Hogan, Paul Resnick, and Ian Soboroff, editors, *Proceedings of the Seventh International Conference on Weblogs and Social Media, ICWSM 2013, Cambridge, Massachusetts, USA, July 8-11, 2013*. The AAAI Press, 2013.

[37] Leo Iaquinta, Marco De Gemmis, Pasquale Lops, Giovanni Semeraro, Michele Filannino, and Piero Molino. Introducing serendipity in a content-based recommender system. In *Proc. HIS*, pages 168–173. IEEE, 2008.

[38] Kensuke Onuma, Hanghang Tong, and Christos Faloutsos. Tangent: A novel, 'surprise me', recommendation algorithm. In *Proc. KDD*, pages 657–666, 2009.

[39] Ido Guy, Roy Levin, Tal Daniel, and Ella Bolshinsky. Islands in the stream: A study of item recommendation within an enterprise social stream. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '15, pages 665–674, New York, NY, USA, 2015. ACM.