Mining Subjective Properties on the Web

Immanuel Trummer*
EPFL
Lausanne, Switzerland
immanuel.trummer@epfl.ch

Alon Halevy
Google, Inc.
Mountain View, USA
halevy@google.com

Hongrae Lee
Google, Inc.
Mountain View, USA
hrlee@google.com

Sunita Sarawagi
Google, Inc. and IIT Bombay
Mountain View, USA/Mumbai, India
sarawagi@google.com

Rahul Gupta
Google, Inc.
Mountain View, USA
grahul@google.com

ABSTRACT

Even with the recent developments in Web search of answering queries from structured data, search engines are still limited to queries with an objective answer, such as EUROPEAN CAPITALS or WOODY ALLEN MOVIES. However, many queries are subjective, such as SAFE CITIES, or CUTE ANIMALS. The underlying knowledge bases of search engines do not contain answers to these queries because they do not have a ground truth. We describe the SURVEYOR system that mines the dominant opinion held by authors of Web content about whether a subjective property applies to a given entity. The evidence on which SURVEYOR relies is statements extracted from Web text that either support the property or claim its negation. The key challenge that SURVEYOR faces is that simply counting the number of positive and negative statements does not suffice, because there are multiple hidden biases with which content tends to be authored on the Web. SURVEYOR employs a probabilistic model of how content is authored on the Web. As one example, this model accounts for correlations between the subjective property and the frequency with which it is mentioned on the Web. The parameters of the model are specialized to each property and entity type.

SURVEYOR was able to process a large Web snapshot within a few hours, resulting in opinions for over 4 billion entity-property combinations. We selected a subset of 500 entity-property combinations and compared our results to the dominant opinion of a large number of Amazon Mechanical Turk (AMT) workers. The predictions of SURVEYOR match the results from AMT in 77% of all cases (and 87% for test cases where inter-worker agreement is high), significantly outperforming competing approaches.

*This work was done while the author was at Google, Inc.

SIGMOD'15, May 31–June 4, 2015, Melbourne, Victoria, Australia.
ACM 978-1-4503-2758-9/15/05.
http://dx.doi.org/10.1145/2723372.2750548.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

Keywords

Text mining; subjective properties; user behavior model

1. INTRODUCTION

In recent years, Web search engines have invested heavily in answering queries with structured data. For example, queries such as WOODY ALLEN MOVIES or AMERICAN PRESIDENTS will yield a display of the appropriate entities. These queries are enabled by large knowledge bases about important entities and properties of these entities. Albeit vast, these knowledge bases are restricted to objective properties of the entities. Hence, queries with subjective properties such as BIG CITIES, SAFE CITIES, or CUTE ANIMALS would not trigger search results from structured data. Considering that queries about subjective properties are very common in the query stream, this implies lost opportunities for offering rich information.

We describe the SURVEYOR system whose goal is to mine from the Web the dominant opinion about whether a particular property applies to entities of a particular type. Given a property, typically expressed as an adjective (e.g., CUTE), a type (e.g., ANIMALS) and a set of entities of that type from the knowledge base, SURVEYOR decides whether the majority of users associate the property with the entity. The purpose is to build a knowledge base of subjective properties and entities. To the best of our knowledge, we are the first to study this problem.

The state-of-the-art approach to building SURVEYOR would be to use an information extraction or natural-language processing method to simply *count* the number of occurrences on the Web in which the property is attributed to the entity and the number of times in which a negative assertion about the property applying to the entity is found. We would then decide if a property applies to an entity based on a count-based estimator such as the majority vote. Such an approach would be appropriate if we had a sufficiently large and unbiased sample of statements about each entity. In practice, this assumption does not hold for the following reasons.

First, the statement sample might be biased because users with one specific opinion are more likely to express them-

selves than the others. For example, users who experience a certain city as being NOT SAFE might be more likely to point that out than other users who experience the same city as SAFE. Therefore, finding more statements claiming that a certain city is NOT SAFE does not necessarily mean that this is the majority opinion. Second, there are many entities and properties for which only very few statements can be found on the Web. This does not mean that we cannot infer the dominant opinion for them. Consider, for example, the property-type combination BIG CITY. Big cities are mentioned more frequently than small cities. A city about which we find no mentions is likely NOT BIG. Furthermore, such correlations are specific to a particular property-type combination.

To overcome all these challenges, we need a model that captures biases and correlations in how users create content on the Web, and therefore enables us to make inferences that go beyond majority voting on assertions. To that end, a critical component underlying SURVEYOR is a probabilistic model of how assertions about properties of entities are generated in Web content. The parameters of the model are instantiated differently for each combination of entity type and property. We use an iterative expectation-maximization approach [9] to infer optimal parameter values automatically for each type and property. Our method is therefore completely unsupervised, enabling us to scale to large numbers of types and properties. Figure 1 illustrates the flow of SURVEYOR.

The contributions of this paper are the following:

1. We introduce the problem of mining subjective properties of entites from the Web, complementing the current capabilities of search engines to answer queries from structured data. We describe the specific challenges that this problem poses, based on an analysis of several use cases.

2. We describe the architecture and implementation of the SURVEYOR system which parses Web snapshots to extract statements involving entities from the knowledge base and subjective properties. SURVEYOR analyzes sets of statements gathered about specific entity-property pairs to derive the probability that the property applies to the entity according to the dominant opinion.

3. We present a probabilistic model of how authors generate content on the Web. Our model is a parametric Bayesian network whose variables capture hidden author biases. We derive an efficient training and inference algorithm whose time complexity is linear in the number of entities and independent of the number of mentions. Thus our probabilistic model is applicable to Web-scale data.

4. We describe our experience of applying SURVEYOR to a Web snapshot of 40 TB, leading to extractions concerning 7 million property-type combinations and resulting in probabilities for over 4 billion entity-property pairs with a processing time of few hours on a large cluster. We evaluate the precision of SURVEYOR by comparing its output to opinions collected from a large number of Amazon Mechanical Turk workers; SURVEYOR outperforms baselines by more than 20 $\sim$ 30% in precision.

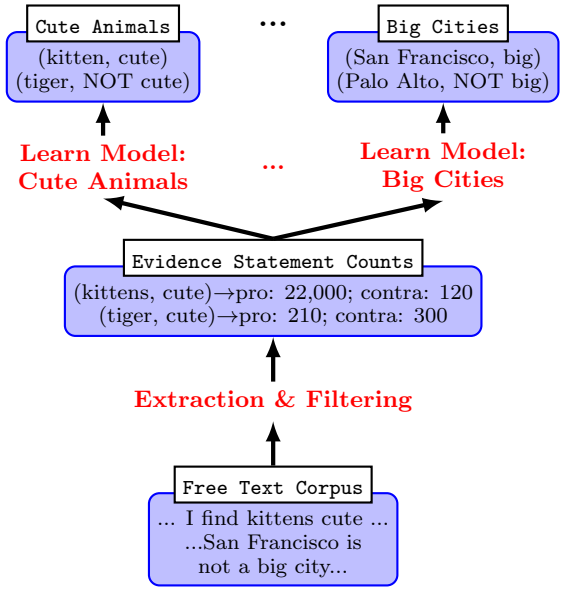


Figure 1: Surveyor begins by extracting positive and negative statements about entity-property pairs and counting the number of each. It then learns and applies a type and property specific probabilistic model to decide whether the counts entail a dominant opinion w.r.t. the property and the entity.

The remainder of this paper is organized as follows. In Section 2 we give a formal problem statement and analyze a representative use case leading to several insights that motivate specific design decisions later. We present the architecture of our system in Section 3. In Section 4 we describe the natural language processing methods by which we detect positive and negative statements. In Section 5 we describe our probabilistic model of user behaviour that is based on the observations from Section 2. We derive equations to calculate optimal parameter values for that model in Section 6. In Section 7 we describe the properties of the full data set generated by our system, and experimentally evaluate our system against baseline approaches. In Section 8 we discuss related work in more detail.

2. PROBLEM STATEMENT

We begin by introducing our terminology and defining the addressed problem. Then, we present an empirical analysis of an example scenario that uncovered many of the challenges we faced and motivated the proposed solution.

Definitions

A *subjective property* in our scenario is an adjective, optionally associated with preceding adverbs. Example properties include CUTE, DENSELY POPULATED, or VERY SMALL. We focus on properties that are subjective, meaning that there is no objective ground truth about whether they apply to certain entities. While different users might disagree on whether subjective properties apply to certain entities, we assume that there is a *dominant opinion* for many entity-property combinations, meaning that a significant part of the user base agrees on whether the property applies to the entity. When being asked whether the property CUTE applies

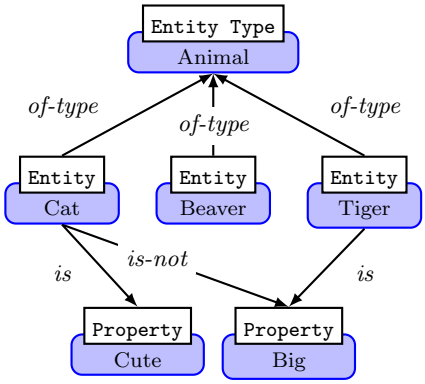


Figure 2: Graphical depiction of problem setting for entity type ANIMAL: typed entities can be connected to subjective properties with positive (is) or negative (is-not) polarity.

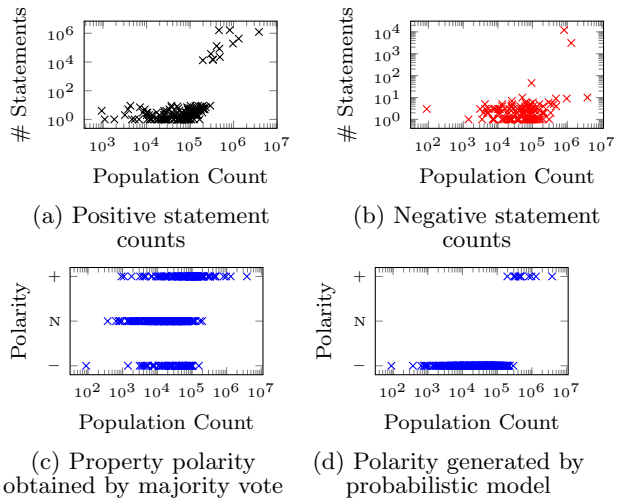


Figure 3: We interpret the statement counts (Figures 3(a)/3(b)) using majority vote (Figure 3(c)) and a probabilistic user model (Figure 3(d)).

to PUPPY or whether BIG applies to LOS ANGELES, without being given any specific context, a majority of users would intuitively answer with YES. In such cases we have a *positive* dominant opinion while otherwise we have a *negative* dominant opinion.

We assume that a *knowledge base* is available which stores entities with their associated entity type (e.g., entity TIGER of type ANIMAL), as well as additional (objective) information on those entities. *Subjective queries* use subjective properties to characterize entities. This requires us to find out which subjective properties the average user associates with which entities in the knowledge base. Upon receipt of a subjective query, the search engine can exploit high-confidence entity-property associations and offer links to supporting content on the Web as query result.

This leads to the *subjective property mining* problem which is defined as follows: Given a document corpus W, an entity e, and a subjective property p, determine the polarity of the dominant opinion on the entity-property pair (e, p) among the authors of W, i.e., whether the majority of authors would intuitively associate the property with the entity. Given a document corpus W and a threshold ρ, Surveyor solves the *all pairs* version of the subjective property mining problem that determines dominant opinions for all pairs (e, p) where entities of the same type co-occur with the property p in at least ρ statements in W. See Figure 2 for a graphical depiction of our problem setting.

The region a user comes from can influence the probability that (s)he associates certain entities with certain properties. For example, Chinese users might have different ideas than American users about what constitutes a BIG CITY. SURVEYOR can produce region-specific results if the input is restricted to Web sites with specific domain extensions. In general, we can specialize the output of SURVEYOR for any user group by restricting the input to documents that have been authored by that group.

Empirical Analysis

We considered a test scenario where we had to decide for 461 Californian cities whether the property BIG applies (additional test scenarios are shown in Appendix A). We collect evidence for a specific city X by issuing Google queries for positive statements of the form "X IS A BIG CITY" and queries for negative statements of the form "X IS NOT A BIG CITY". Note that in our actual system we used a more sophisticated Natural Language Processing (NLP) approach to extract the statements, which recognizes a much broader class of patterns.

We checked the names of all cities that received at least 100 hits in total for ambiguity and discarded 11 out of 23 of those cities since the top search result did not return the city that the population count refers to. We conclude that disambiguation is crucial in our scenario; our extraction mechanism that we present in Section 4 uses annotated Web documents as input that have been pre-processed by an entity tagger using state-of-the-art means for disambiguation.

Unlike our final evaluation of SURVEYOR with Mechanical Turk, in this exploration we use the population of the city as a proxy for its size. We chose this example because population correlates with the property BIG, but in general, the property will not necessarily be correlated to knowledge in the knowledge base. Figures 3(a) and 3(b) show the number of positive and negative statements for all cities that passed the ambiguity test, ordered by population count. There are many cities for which we find a non-negligible number of positive and negative statements at the same time. This means that a significant fraction of users disagrees with the dominant opinion. Therefore, we must take into account *subjectivity* and must aggregate contradictory results.

The first approach that comes to mind for aggregating contradictory results is to take the majority vote. Under the assumption that we find sufficiently large and unbiased sets of statements about each entity, the majority vote is representative for the dominant opinion. Figure 3(c) shows the result of the majority vote: for each city, we compare the number of positive statements with the number of negative statements and mark it as BIG (polarity=+) if the number of positive statements is higher; we mark it as NOT BIG (polarity=−) if the number of negative statements is higher. We mark polarity=N if both counters are equal. The figure shows that we find more negative than positive statements for some cities. Such cities should probably be marked as

NOT BIG which underscores the need to use NLP technology that can distinguish positive from negative assertions, and excludes the use of purely occurrence-based approaches that have been used by prior work in product tagging [2, 4, 5].

Despite the fact that we distinguish negative and positive statements, the result quality in Figure 3(c) seems rather low. Many cities are marked as big despite a relatively low population count. Looking for an explanation, we find that the total number of negative statements (see Figure 3(b)) is much lower than the number of positive statements (see Figure 3(a)). We have no reason to suspect that our sample contains mainly big cities (the opposite is probably true given the population counts). This indicates that counters of positive and negative statements can follow different probability distributions, i.e. we might have a *polarity bias*.

In addition to the lack of a correlation between property polarity and population counts, the results in Figure 3(c) are poor for another reason: for many cities, we cannot decide whether they are big or not because we could not extract positive or negative statements about them. At first glance, this problem seems hard to solve. Looking again at Figures 3(a) and 3(b), we realize however that the total number of positive and negative statements is correlated with the population count. Big cities tend to be mentioned more often on the Web than small cities. If we are able to automatically detect such correlations, we can infer that a city that is never mentioned on the Web is not big with high probability. Note that the lack of mentions for an entity is rather meaningless when working with a small collection of documents. But when considering the entire Web, finding no occurrences of an entity means that billions of content-generating users made the decision not to mention the entity; at sufficiently large scale, the lack of any evidence can be evidence as well. We conclude that taking into account *occurrence bias*, the correlation between property polarity and occurrence frequency, can be helpful.

Our hypothesis is that it is possible to account all types of bias by using a probabilistic user behavior model. Figure 3(d) shows the result of applying an early version of the probabilistic model presented in Section 5 to interpret the statement counts from Figures 3(a) and 3(b). The result quality is markedly better than in Figure 3(c): we obtain a decision for each city and polarity is strongly correlated with population count.

As we have seen, users are more likely to write about cities if they associate the property BIG with them. This implies that users are *less* likely to mention cities if they associate the property SMALL with them, as big and small are antonyms. This means that the occurrence bias for those two properties is different. In general, we observed that all types of bias mentioned before may differ across properties but also across entity types. We conclude that polarity bias and occurrence bias do *not generalize* and must be inferred separately for different property-type combinations.

The method that we present in the following sections is based on all those observations.

3. SYSTEM OVERVIEW

Algorithm 1 provides a high-level description of SURVEYOR. The input is a collection of annotated Web documents W, from which we extract evidence, a knowledge base KB containing entities with their types, and a threshold parameter ρ whose semantics we discuss later. The output of Algorithm 1 is a set of tuples assigning entity-property combinations to a polarity representing the dominant opinion.

Algorithm 1 Surveyor algorithm

```
1: // W: Web snapshot; KB: Knowledge base
2: // ρ: occurrence threshold
3: function SURVEYOR(W, KB, ρ)
4:     Iterate over documents in W to extract evidence
5:     for ⟨type, property⟩ : at least ρ extractions do
6:         Learn model parameters
7:         for entity ∈ KB : entity is of type do
8:             Calculate prb = Pr(property applies)
9:             Add ⟨entity, property, +⟩ to result if prb > 1/2
10:            Add ⟨entity, property, −⟩ to result if prb < 1/2
11:        end for
12:    end for
13:    return result
14: end function
```

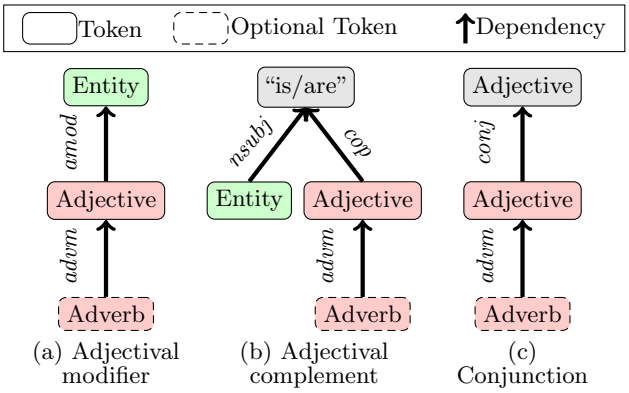
Extracting Evidence. We extract positive and negative evidence from a text corpus containing a snapshot of the Web. The corpus text was preprocessed using NLP tools and contains annotations mapping text mentions of entities to our knowledge base which is an extension of Freebase. We consider each mention of a knowledge base entity and analyze the surrounding free text. We use several patterns to detect evidence statements and to extract the properties that they connect to the entity and the polarity of the association. Section 4 provides more details on the extraction phase.

Next, we group evidence by the entity-property pair it refers to. For each pair, we compute two counters: the total number of positive statements and the total number of negative statements. Those counters are the input for the next step of the algorithm.

Evidence Interpretation. We interpret the collected evidence to calculate the probability that a specific property applies to a given entity. The knowledge base associates each entity with an entity type (the knowledge base may actually associate multiple types with an entity but we use only the most notable type). We group evidence by entity type, and aggregate for each property-type combination the total number of extracted statements. We only compute probabilities for property-type combinations for which the number of extracted statements is sufficiently high.

We consider each property-type combination separately. For each combination, we first use the collected evidence to instantiate a probabilistic model of how authors decide to issue positive or negative statements. Section 5 describes the model and justifies the underlying assumptions. The model is parameterized and we show in Section 6 how to learn optimal parameter values in an iterative approach. The instantiated model can be used to determine for a given entity the probability that the dominant opinion on it is positive, based on the collected evidence. We calculate probabilities for all entities in the knowledge base that belong to the current entity type. In paritcular, this includes entities for which no evidence was extracted at all. For some property-type pairs, SURVEYOR may draw conclusions for these unmentioned entities as well.

We currently assume a positive dominant opinion if the probability is greater than 0.5, and a negative dominant



(a) Adjectival modifier (b) Adjectival complement (c) Conjunction

Figure 4: Evidence extraction patterns as Stanford typed dependency trees: property tokens are red.

Table 1: Example extractions

Statement	Pattern	Entity	Property
Snakes are dangerous animals	Adjectival modifier	snake	dangerous
Chicago is very big	Adjectival complement	Chicago	very big
Soccer is a fast and exciting sport	Conjunction	soccer	exciting

opinion if it is less than 0.5. We can chose a different threshold if we want to trade precision for recall.

4. EXTRACTING EVIDENCE

In this section we describe some of the details of evidence extraction in SURVEYOR. In particular, we describe our extraction patterns, how we limit the extraction to intrinsic evidence, and how we determine the polarity of evidence.

An evidence statement connects an entity to a property; a positive statement claims that the property applies to the entity, a negative statement claims the opposite. We recall that in our setting, a property is an adjective, optionally associated with adverbs (e.g., DENSELY POPULATED, VERY BIG). The input for evidence extraction is an annotated Web snapshot that was preprocessed using NLP tools similar to the Stanford parser[1] and by an entity extractor that identifies mentions of knowledge base entities using disambiguation techniques. The annotations contain the resulting dependency tree representation of sentences and the links to knowledge base entities. We consider sentences in our text corpus that mention at least one entity from the knowledge base and analyze their dependency parse trees.

We manually devised several patterns against which we match the sentences in the corpus. Figure 4 shows a simplified representation of those patterns. We obtain the properties associated with an entity by matching the corresponding pattern with the dependency tree. The tokens that together form the property are colored in red in Figure 4. Table 1 shows an example statement for each of the three patterns and the corresponding extractions. In the first example, note that ANIMALS is coreferential with SNAKES, i.e. those are two mentions of the same entity. Property DANGEROUS is adjectival modifier of the second mention. Also note that the third statement contains two patterns: using the adjectival modifier pattern we can also extract property FAST for entity SOCCER.

Our patterns do not explicitly target subjective properties (e.g., we might extract objective properties such as AMERICAN for a city), but most extracted properties turn out to be subjective in practice. The patterns depicted in Figure 4 are relatively restrictive. We tried several variations of those patterns (see Appendix B), allowing for instance an extended set of verbs for the top node of the adjectival complement pattern depicted in Figure 4(b), but came to the conclusion that the patterns in Figure 4 offer the best tradeoff between precision and recall for our application. In general, our design of the extraction and filtering mechanisms prioritizes precision over recall since we apply SURVEYOR to large input document collections where recall is less critical.

Once an evidence statement is extracted, we apply several heuristics to filter out non-intrinsic evidence statements. An example of a non-intrinsic statement is NEW YORK IS BAD FOR PARKING because it only refers to a specific aspect of the city (in contrast to the statement NEW YORK IS BAD). While it would be possible to use expressions such as BAD FOR PARKING as properties, we assume that the number of statements found for such complex properties would be too low to allow reliable inference. In order to recognize non-intrinsic statements that refer only to a specific aspect of an entity, we search for sub-trees in the dependency tree that could represent constrictions. We search for sub-trees that have a specific position in the dependency tree relative to the detected pattern (e.g., additional sub-trees of the top level node in Figure 4(b)) and contain nodes with specific labels (e.g., labels indicating prepositions). If such a sub-tree is found, we assume that the statement is non-intrinsic. While this filtering mechanism can be rather conservative at times, we found it to improve precision significantly.

As another example, compare the statements SOUTHERN FRANCE IS WARM and GREECE IS A SOUTHERN COUNTRY. Both statements use the adjective SOUTHERN as adjectival modifier for an entity of type COUNTRY. They differ in that the first statement uses the adjective to refer to a specific part of a country (France) while the second statement claims that SOUTHERN is an intrinsic property of an entire country (Greece). To filter out non-intrinsic statements that refer to a part of an entity (instead of distinguishing the entity from other entities of the same type), we require sentences in which the adjectival modifier pattern was detected to be coreferential. Note that this test distinguishes the two example statements given above.

Next, we determine the polarity of the statement by exploiting annotations in the dependency tree that indicate negations: for example, the sentence I DON'T THINK THAT SNAKES ARE NEVER DANGEROUS contains two negations (the negations DON'T and NEVER). Figure 5 shows the dependency tree representation of that sentence (negations are marked in red). Note that this sentence is recognized as evidence since a sub-tree (SNAKES ARE DANGEROUS) matches the adjectival complement pattern from Figure 4.

We decide the polarity by following the path in the dependency tree from the property token to the root: starting

[1] `http://nlp.stanford.edu/software/stanford-dependencies.shtml`

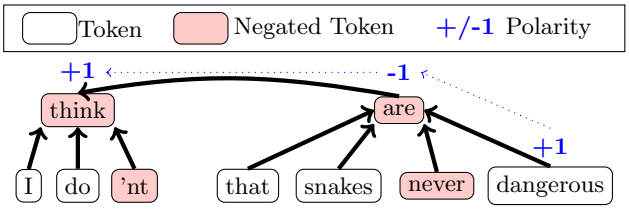


Figure 5: We detect the polarity of the example statement by counting the negated tokens on the path from property (dangerous) to tree root (think).

with a polarity of +1, we change the sign every time we encounter a negated token on that path (a negated token has a negation as child element). If the number of negated tokens is odd, then the polarity of the statement is negative, and otherwise it is positive. Note that this method recognizes even double negations which are rare but do sometimes appear for the properties we consider.

We considered taking into account antonym relationships between adjectives when identifying negations, e.g., interpreting the statement PALO ALTO IS SMALL as negation of PALO ALTO IS BIG. We decided against it for two reasons: First, even if two adjectives are registered as antonyms (e.g., within a database such as WordNet), they usually do not represent the exact opposite of each other. Users who consider a city as NOT BIG do not necessarily consider it SMALL. Second, we consider adverb-adjective combinations for which it is often impossible to find any antonyms at all.

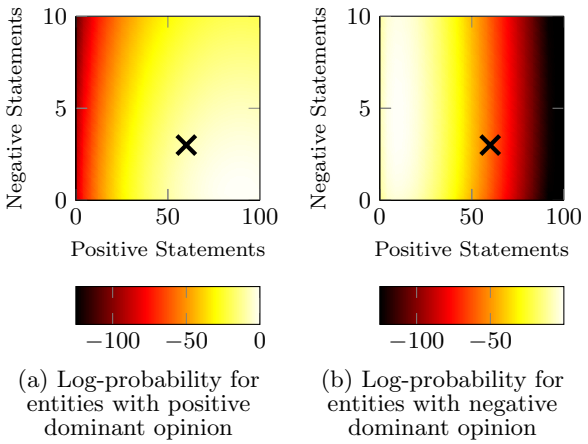
5. MODELING USER BEHAVIOR

As we described in Section 2, the simple approach to estimating the probability of the dominant opinion based on majority vote counting does not work very well because it fails to model the different types of bias that underlie authoring on the Web. To address this fundamental challenge, SURVEYOR employs a probabilistic model that explicitly accounts for behavior of authors on the Web and takes into consideration the different types of biases. This section describes the model, beginning with an overview. Section 6 describes how we learn the model parameters for specific property-type combinations.

5.1 Model Overview

We assume in the following that one specific property-type combination is considered. The output of the extraction stage, described in Section 4, is an evidence tuple $\langle C_i^+, C_i^- \rangle$ for each entity; the evidence consists of the total count of positive statements, C_i^+, and the total count of negative statements, C_i^-, gathered during extraction about the entity and the current property.

Our probabilistic model assumes that each evidence tuple was drawn from one of two possible probability distributions: in the first distribution, we assume that the dominant opinion applies the property to the entity, whereas in the second, the dominant opinion does not. If we know how to express those two probability distributions then we can calculate for each evidence tuple the probability with which it was drawn from one distribution or the other; this is at the same time the probability that the entity to which the evidence tuple refers, does or does not have the current property.



(a) Log-probability for entities with positive dominant opinion

(b) Log-probability for entities with negative dominant opinion

Figure 6: Each property-type combination is associated with two probability distributions over the statement counters: the dominant opinion on the entity for which we receive the counts $\langle 60, 3 \rangle$, marked by X, is more likely to be positive.

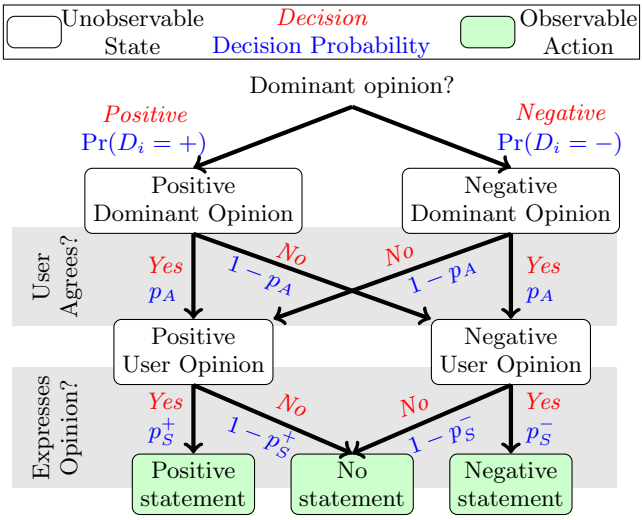


Figure 7: Probabilistic user behavior model: users agree with the dominant opinion on specific entity-property pairs with probability p_A; they state their opinion with probability p_S^+ if it is positive and probability p_S^- if it is negative.

EXAMPLE 1. *Figure 6 illustrates the scenario described above: we have two two-dimensional probability distributions that assign tuples of positive and negative statement counts to probabilities, lighter colors represent higher probabilities in the figure. Assume we receive the evidence tuple $\langle 60, 3 \rangle$ for an entity (this point is marked by the black X in Figure 6). As the distribution for entities with positive dominant opinion (depicted in Figure 6(a)) assigns a higher probability to that tuple than the distribution for entities with negative dominant opinion (depicted in Figure 6(b)), the current property is more likely to apply to that entity than not.*

Statements are issued by users who have a certain opinion about an entity and decide to express that opinion on the

Web. In order to model the probability to receive a certain number of positive or negative statements, we must model the probability that a single user decides to issue a positive or negative statement.

Figure 7 is a graphical depiction of our user model. We denote the dominant opinion for entity i of the given type by D_i. The top node, the dominant opinion about an entity-property pair, is what we are trying to compute. However, the only observable variables are the green rectangles at the bottom, namely, the actual statements (positive and negative) that we can extract from the text corpus. The probabilistic model contains internal variables that model how actual statements on the Web are created relative to a given dominant opinion.

First, the dominant opinion could be either positive or negative, which is represented by the two rectangles below the dominant opinion. We denote the probability for a positive dominant opinion by $\Pr(D_i = +)$ and the probability for a negative dominant opinion by $\Pr(D_i = -)$. Those probabilities are initially unknown but the model enables us to calculate them.

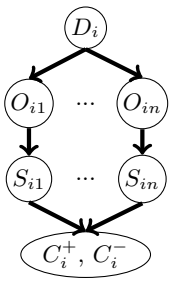
Second, given the polarity of the dominant opinion, an author of a web page may or may not agree with it. This connects to our observation from Section 2 that subjectivity plays an important role in our scenario. We will denote by p_A the probability that an author agrees with the dominant opinion, and therefore the transition from the second layer of Figure 7 to the third layer is done with probabilities p_A and $1 - p_A$, depending on the direction.

Finally, we model the probability that an author will actually write a statement about whether the property applies to the entity. We have observed various types of bias in Section 2, leading us to believe that the probability for writing a statement may depend on the user opinion. We therefore introduce two separate parameters, p_S^+ and p_S^-, representing the probability that a user mentions an entity with which (s)he associates the current property (p_S^+) and the probability that a user mentions an entity with which (s)he does not associate the current property (p_S^-).

We note that the model is parameterized per each property-type combination, as opposed to, for instance, all types-property, or all types-all properties pairs. This choice is based on our observation (in Section 2) that the right settings for the parameters vary across entity types for a fixed property but also across properties for a fixed type. Our experiences when collecting ground truth data for our experimental evaluation, described in Section 7.3, further support this design decision.

5.2 Model Details

The process described in the previous subsection can be expressed as a Bayesian network as illustrated in Figure 8. We use random variables D_i, C_i^+ and C_i^- to model the dominant opinion on the i-th entity, the number of positive statements and the number of negative statements collected for the i-th entity. We conceptually introduce two variables, O_{iw} and S_{iw}, for the Web document with index $w \in \{1, \dots n\}$. Variable O_{iw} represents the opinion of the corresponding author about whether the current property applies to the i-th entity; this opinion may be positive or negative (represented by $O_{iw} = +$ and $O_{iw} = -$ respectively), it is consistent with the dominant opinion (i.e., $O_{iw} = D_i$) with probability p_A and inconsistent with probability $1 - p_A$. Variable S_{iw} captures whether the author of document w decided to make a positive or negative or no statement about whether the current property applies to entity i; those three cases are represented by $S_{iw} = +$, $S_{iw} = -$, or $S_{iw} = N$ respectively. The probability of making a statement is p_S^+ if $O_{iw} = +$ and p_S^- if $O_{iw} = -$. The counter variable C_i^+ counts the number of documents w such that $S_{iw} = +$ and variable C_i^- counts the number of documents such that $S_{iw} = -$. Note that the values for the parameters p_A, p_S^+, and p_S^- are fixed for each property-type combination. We illustrate those definitions by an example.



$\Pr(O_{iw}\|D_i)$	D_i +	D_i −
O_{iw} +	p_A	$1-p_A$
O_{iw} −	$1-p_A$	p_A

$\Pr(S_{iw}\|O_{iw})$	O_{iw} +	O_{iw} −
S_{iw} +	p_S^+	0
S_{iw} N	$1-p_S^+$	$1-p_S^-$
S_{iw} −	0	p_S^-

Figure 8: Probabilistic model as Bayesian network with conditional probabilities

Example 2. *Assume we want to find out which* animals *are considered* cute. *Let* kitten, dog, spider *be the list of entities (animals) that we consider. Then D_i for $1 \le i \le 3$ represents the dominant opinion about whether the i-th animal in that list is* cute, *counter C_i^+ represents the number of times the i-th animal was mentioned as being* cute *on the Web, and C_i^- is the number of times the animal was mentioned as being* not cute. *If $O_{1w} = +$ and $S_{1w} = N$, for some Web site w, then the author of that site considers* kittens *as* cute *but did not decide to express that opinion.*

Assume that there is generally a high agreement between users whether specific animals are cute or not. Then we expect to obtain a relatively high agreement parameter p_A for this property-type combination from the parameter learning algorithm presented in Section 6. Users are more likely to state the fact that they find a specific animal cute *than the fact that they consider an animal* not cute *and this is why we expect to obtain statement probabilities such that $p_S^+ \gg p_S^-$. Note that it might be the inverse for other property-type combinations: for instance, users might have a bias towards rather expressing their opinion if they consider a city* not safe *than if they consider it* safe. *Therefore, we expect to infer parameter values such that $p_S^- \gg p_S^+$ for property-type combination* safe cities.

Our goal now is to estimate the distribution over the variable D_i given the observed counts, i.e., $\Pr(D_i|C_i^+, C_i^-)$. We elaborate on how to compute $\Pr(D_i|C_i^+, C_i^-)$ with respect to the Bayesian network in Figure 8. First note that $\Pr(D_i|C_i^+, C_i^-) \propto \Pr(C_i^+, C_i^-|D_i)\Pr(D_i)$. Since we wish to be agnostic about the prior probability of D_i we set $\Pr(D_i = +) = \Pr(D_i = -) = 0.5$. Since the counts C_i^+, C_i^- are deterministic functions of S_{iw}, we first solve for $\Pr(S_{iw}|D_i)$.

From the Bayesian network, we obtain

$$(S_{iw}|D_i) = \sum_{O_{iw} \in \{+,-\}} \Pr(S_{iw}|O_{iw}) \Pr(O_{iw}|D_i).$$

By substituting, $D_i = +$ or $-$ and $S_{iw} = +$ or $-$, we get four possibilities as follows:

$$\Pr(S_{iw} = +|D_i = +) = p_A \cdot p_S^+$$
$$\Pr(S_{iw} = -|D_i = +) = (1 - p_A) \cdot p_S^-$$
$$\Pr(S_{iw} = +|D_i = -) = (1 - p_A) \cdot p_S^+$$
$$\Pr(S_{iw} = -|D_i = -) = p_A \cdot p_S^-$$

The variables C_i^+, C_i^- are obtained by summing up n variables S_{iw} each of which can be $+$,$-$, or neutral. We assume that the variables S_{iw} are independent for different w since the chances that two randomly selected documents on the Web are authored by the same person are negligible. This implies that (C_i^+, C_i^-) follows a Multinomial distribution where $\Pr(C_i^+ = a, C_i^- = b|D_i = +) =$

$$\frac{n!}{a!b!(n-a-b)!}(p_+^+)^a(p_+^-)^b(1 - p_+^+ - p_+^-)^{n-a-b}$$

where p_+^+ and p_+^- denote $\Pr(S_{iw} = +|D_i = +)$, and $\Pr(S_{iw} = -|D_i = +)$ respectively (i.e. the subscript is the dominant opinion and the superscript is the user opinion).

We can approximate this multinomial distribution as a product of two Poisson distributions since n is expected to be very large compared to both (C_i^+, C_i^-) [14, 18] since for any entity and property, the number of Web sites on which they appear together is very small compared to the total number of Web sites. Thus, we can rewrite the above expression as $\Pr(C_i^+ = a, C_i^- = b|D_i = +) = \Pr(C_i^+ = a|D_i = +) \Pr(C_i^- = b|D_i = +) =$

$$\text{Pois}(a; \lambda_+^+ = np_+^+)\text{Pois}(b; \lambda_+^- = np_+^-)$$

We derive a similar expression for the case where $D_i = -$.

In summary, we obtain four different Poisson distributions that can be described using the three parameters p_A, p_S^+, and p_S^-: the two counter distributions for positive and negative statements for entities to which the current property applies according to the dominant opinion, and the two corresponding distributions for entities to which the property does not apply. Each of the four distributions is described by one of four Poisson parameters $\lambda_{\sigma 1}^{\sigma 2}$ with $\sigma_1, \sigma_2 \in \{+, -\}$ where the subscript σ_1 refers to the dominant opinion and the superscript σ_2 to the statement polarity such that $(C_i^{\sigma 2}|D_i = \sigma 1) \sim \text{Pois}(\lambda_{\sigma 1}^{\sigma 2})$. The following equations express the Poisson parameters in terms of the three model parameters:

$$\lambda_+^+ = n \cdot p_A \cdot p_S^+ \qquad \lambda_+^- = n \cdot (1 - p_A) \cdot p_S^-$$
$$\lambda_-^- = n \cdot p_A \cdot p_S^- \qquad \lambda_-^+ = n \cdot (1 - p_A) \cdot p_S^+$$

EXAMPLE 3. *Assume we choose $p_A = 0.9$, $np_S^+ = 100$, and $np_S^- = 5$. The agreement parameter is relatively high and p_S^+ significantly higher than p_S^-; those are the characteristics of Example 2. We obtain $\lambda_+^+ = 90$, $\lambda_+^- = 0.5$, $\lambda_-^- = 4.5$, and $\lambda_-^+ = 10$. The corresponding distributions over the evidence tuples are the ones shown in Figure 6.*

Algorithm 2 Learning model parameters

```
1: // E: All evidence about one property-type pair
2: // X: number of iterations
3: function EM(E, X)
4:     Guess initial parameter vector θ_0
5:     for k ← 1 to X do
6:         Calculate opinion probabilities Pr(D|E, θ_{k-1})
7:         Calculate θ_k using opinion probabilities
8:     end for
9:     return θ_X
10: end function
```

6. CALCULATING PARAMETER VALUES

The probabilistic model introduced in the last section contains at the same time unknown parameters (p_A, p_S^+, and p_S^-) and unobservable random variables (the dominant opinion D_i on each entity). Knowing all parameter values would allow to infer the dominant opinion on each entity from the counts of positive and negative statements. Knowing the dominant opinion on each entity would allow to calculate optimal parameter values from the statement counts. We know however neither parameter values nor the dominant opinion and adopt therefore an iterative expectation-maximization (EM) approach [9].

Algorithm 2 shows a high-level overview of the expectation-maximization approach, applied to our scenario. The function represented in Algorithm 2 uses the evidence E that was collected about one specific property-type combination as input, i.e. $E = \{\langle c_i^+, c_i^- \rangle | i = 1..m\}$ is the count of positive and negative statements concerning the current property for all entities of the current type. The output is a three-dimensional vector θ_X containing near-optimal values for the three model parameters. The algorithm is iterative and executes X times the following two steps. First, it uses the estimates for the parameter values derived in the last iteration (or a vector of default values for the first iteration) to calculate probabilities for the dominant opinion about each entity. Second, it uses the opinion probabilities to calculate the most likely values for the parameters. We provide details on the second step in this section; the last section sketched how opinion probabilities can be calculated. Note that Algorithm 2 is executed separately for each property-type pair.

The EM approach is an extension of the Maximum- Likelihood (ML) method [9]. The goal of the ML method is to maximize a likelihood function $L(\Theta)$ in the parameter values Θ. The likelihood function represents the probability of obtaining given observations when assuming specific parameter values. The EM approach introduces a likelihood function that depends not only on the parameters but also on the unobservable random variables. In our scenario, this likelihood function $L(d, \Theta) := \Pr(E, D = d|\Theta)$ depends on the parameters $\Theta = \langle p_A, p_S^+, p_S^- \rangle$ but also on the values $d = \langle d_1, \ldots, d_m \rangle$ for the vector $D = \langle D_1, \ldots, D_m \rangle$ of unobservable dominant opinions. Algorithm 2 calculates a probability distribution over D in the first step of each iteration. Following the EM approach, we use that distribution in the second step to calculate function $Q_k(\Theta)$, representing the expected value of the logarithm of the likelihood function in the k-th iteration:

$$Q_k(\Theta) = \sum_{d \in \{+,-\}^m} \Pr(D = d|\theta_{k-1}, E) \log L(d, \Theta) \qquad (1)$$

We choose the new parameter estimate by maximizing Q_k:

$$\theta_k = \arg\max_{\theta}(Q_k(\theta)).$$

Note that the formula for Q_k contains the variable Θ, in which we maximize, as well as the constant vector θ_{k-1} that was calculated in the last iteration. In the following, we outline how Q_k can be expressed and maximized. Evaluating (1) directly is inefficient since the sum has an exponential number of terms in the number of entities. By summarizing the corresponding terms, we obtain a formula for Q_k with a linear number of terms in the number of entities:

$$\sum_{i=1}^{m} \sum_{d_i \in \{+,-\}} \Big[\log(\Pr(D_i = d_i, E_i|\theta)) \Pr(D_i = d_i|\theta_{k-1}, E_i) \Big]$$

We use $r_i^+ := \Pr(D_i = +|\theta_{k-1}, E_i)$ to denote the probability that the dominant opinion on the i-th entity is positive. Maximizing Q_k is equivalent to maximizing Q'_k which is obtained by neglecting constant factors in Q_k and yields:

$$\begin{aligned} Q'_k(\Theta) = & \\ & \sum_i [r_i^+ (c_i^+ \log(\lambda_+^+) - \lambda_+^+ + c_i^- \log(\lambda_+^-) - \lambda_+^-) \\ & + (1 - r_i^+)(c_i^+ \log(\lambda_-^+) - \lambda_-^+ + c_i^- \log(\lambda_-^-) - \lambda_-^-)] \end{aligned}$$

Details of the transformation from Q_k to Q'_k are given in Appendix C. We can find the maximum of Q'_k by setting the partial derivatives $\partial Q'_k/\partial p_A$, $\partial Q'_k/\partial p_S^+$, and $\partial Q'_k/\partial p_S^-$ to zero. In our implementation, we speed up computations by trying a fixed set of values for p_A and maximizing Q'_k in p_S^+ and p_S^- for each value of p_A by setting the partial derivatives for p_S^+ and p_S^- to zero. We introduce several short notations to be able to conveniently express the formulas for p_S^+ and p_S^- that maximize Q'_k for fixed value of p_A. By $g_{\sigma 1}^{\sigma 2}$ with $\sigma_1, \sigma_2 \in \{+,-\}$, we denote the estimated number of statements with polarity σ_2 for entities with property polarity σ_1 according to the dominant opinion (e.g., g_+^- is the estimated number of negative statements about positive entities). By g_+ we denote the estimated number of entities with positive dominant opinion and by g_- the number of entities with negative dominant opinion:

$$\begin{aligned} g_+^+ &= \sum_i (c_i^+ r_i^+) & g_+^- &= \sum_i (c_i^- r_i^+) \\ g_-^+ &= \sum_i (c_i^+ (1 - r_i^+)) & g_-^- &= \sum_i (c_i^- (1 - r_i^+)) \\ g_+ &= \sum_i r_i^+ & g_- &= \sum_i (1 - r_i^+) \end{aligned}$$

Now we can finally express the formulas for np_S^+ and np_S^- (we prefer working with np_S^+ and np_S^- over working with p_S^+ and p_S^- in our implementation to minimize rounding errors) that maximize Q'_k and Q_k for fixed values of p_A:

$$\begin{aligned} np_S^+ &= (g_+^+ + g_-^+)/(g_- + p_A g_+ - p_A g_-) \\ np_S^- &= (g_+^- + g_-^-)/(g_+ + p_A g_- - p_A g_+) \end{aligned}$$

We derived expressions for the parameters values that can be evaluated in $O(m)$, i.e. in linear time in the number of entities. This allows SURVEYOR to scale to large entity sets as demonstrated in the next Section.

7. EXPERIMENTAL EVALUATION

We used SURVEYOR to solve the all common pairs version of the subjective property mining problem. Our input corpus was a snapshot of the entire Web. Section 7.1 quantifies input and output data set sizes as well as processing times for the different processing steps. Section 7.2 presents statistics concerning the number of extracted statements for different entities, properties, and types. We selected a limited set of entity-property combinations as sample to evaluate the precision of SURVEYOR in comparison to simpler approaches. Evaluating precision requires us to establish some kind of ground truth to compare against. Our ground truth is the dominant opinion among Web users about whether certain subjective properties apply to certain entities; we can approximate that dominant opinion by asking a sufficient number of *Amazon Mechanical Turk* (AMT) workers. Section 7.3 describes the test samples and how we approximate the dominant opinion on them using AMT. Section 7.4 compares SURVEYOR with alternative approaches against AMT. We summarize and discuss all results in Section 7.5.

7.1 Mining Subjective Properties on Web Scale

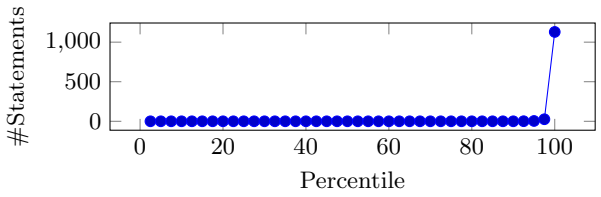
The data processing pipeline was executed on up to 5000 nodes. We used an annotated Web snapshot of 40 TB as input. Extracting evidence statements from that snapshot took around one hour. We thereby extracted over 922 million evidence statements concerning over 60 million entity-property combinations. Our knowledge base is an extension of Freebase; extracting the relevant information from the knowledge base (entities with their most notable types) took around 20 minutes. Combining information obtained from the knowledge base with evidence extracted from the Web snapshot and grouping entities by type took around one hour. The grouping of 60 million entity-property pairs by type, yielded 7 million distinct property-type pairs. We filtered this set down to 380,000 by removing the property-type pairs with fewer than 100 evidence sentences. On each property-type pair we ran the EM algorithm of Section 6 to get dominant opinions for 4 billion entity-property pairs subsumed by the 380,000 type-property pairs. The total time for this step was only 10 minutes. We attribute the efficiency of the EM step to the closed-form expressions we derived for each of the E and M steps.

7.2 Extraction Statistics

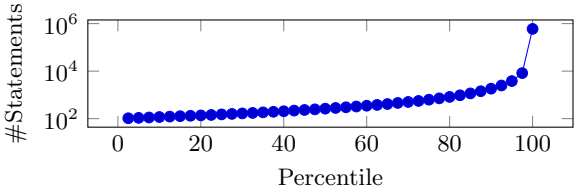
Figure 9 shows statistics concerning the number of extracted statements for entities, properties, and types. Figure 9(a) shows percentiles for the number of statements extracted about specific knowledge base entities: the 20th percentile shows for instance the number of statements such that for 20% of all knowledge base entities at most that many statements were extracted per entity. All percentiles up to the 95th percentile are close to zero. This means that most entities are rarely mentioned while few popular entities are the subject of most extracted statements.

Figure 9(b) shows the distribution of extracted statements over different property-type combinations. The distribution is skewed, meaning that a big part of all extracted statements concerns a few popular property-type combinations.

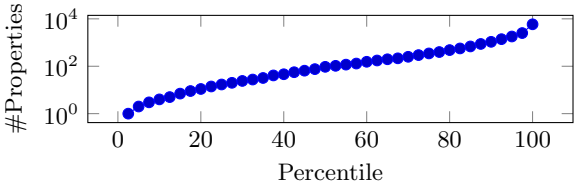
As discussed before, we only consider property-type combinations for which at least 100 statements are extracted. Figure 9(c) shows the distribution of considered properties over the entity types. The distribution is skewed again,



(a) Number of statements extracted about specific knowledge base entities



(b) Number of statements extracted about specific property-type combinations



(c) Number of properties for which more than 100 statements are extracted for specific types

Figure 9: Extraction statistics: a big part of all extracted statements refers to a small set of popular entities and popular property-type combinations; few types are associated with many properties.

meaning that many properties are considered for few types while few properties are considered for most types.

7.3 Test Cases

We evaluate the accuracy of our system against a test dataset containing dominant opinions gathered from AMT workers. Our test dataset comprised 500 entity-property pairs: we selected five entity types and for each type 20 entities and five subjective properties as summarized in Table 2. We selected diverse types and properties that are common in the query stream. We safeguard against undesirable bias that this selection might introduce by reporting on a second series of evaluations over randomly selected entity-property pairs in Appendix D.

Since the goal of SURVEYOR is to find the dominant opinion, we tapped the crowd to collect ground truth data. We approximate the dominant opinion by asking AMT workers: we asked 20 workers about each of the 500 entity-property combinations such that we collected in total 10,000 opinions. We payed a fixed amount of 10 cents for providing an opinion on all entities of the same type for one specific property such that the total cost was 50 US dollars.

Figure 10 shows the results that we obtained from AMT for the property-type combination CUTE ANIMALS: for each animal, we report the number of AMT workers that would associate the property CUTE with it (out of 20 workers in total). Even if CUTE is a subjective property, we see that there is often a strong agreement between workers about

Table 2: Evaluated property-type combinations

Entity Type	Properties
Animals	dangerous, cute, big, friendly, deadly
Celebrities	cool, crazy, pretty, quiet, young
Cities	big, calm, cheap, hectic, multicultural
Professions	dangerous, exciting, rare, solid, vital
Sports	addictive, boring, dangerous, fast, popular

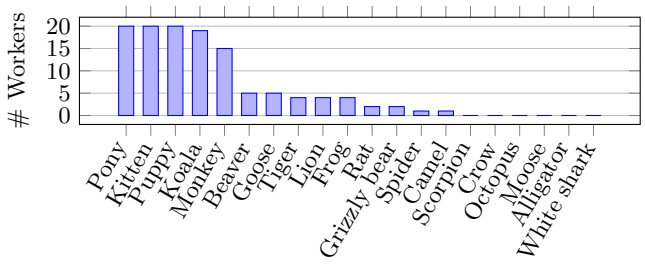


Figure 10: Test case example: how many out of 20 AMT workers call the animal "cute"?

whether specific animals are considered cute or not. In Figure 11 we show the distribution of the worker agreement values over all 500 test cases. We calculated worker agreement as the number of AMT workers that share the same opinion. We observe that there is indeed a dominant opinion that is worth searching for. Averaged over all 500 test cases, we had a worker agreement of 17 out of 20 with almost 180 cases enjoying perfect agreement. Only for 4% of the cases we got ties. We removed these cases from our test set.

While overall agreement was high, we noticed differences across different types and properties. Worker agreement was for instance higher when deciding which ANIMALS are DANGEROUS (average agreement: 18) than when deciding which SPORTS are DANGEROUS (average agreement: 16). Also, people agree more on which SPORTS are DANGEROUS than on which SPORTS are BORING (average agreement: 15). This means that the agreement parameter p_A should be chosen differently for each of the three combinations DANGEROUS ANIMALS, DANGEROUS SPORTS, and BORING SPORTS. This justifies our design decision of treating different property-type combinations separately.

7.4 Experimental Results

We evaluate SURVEYOR on the test cases presented in Section 7.3 by comparing its output to the opinions we collected from AMT. For a given test case, SURVEYOR will either assign positive polarity, negative polarity, or not generate any output (if we calculate a probability of 0.5 for the dominant opinion being positive). We consider the test case unsolved in the latter case. We use three measures to evaluate: coverage, precision, and F1. *Coverage* is the ratio of solved test cases to test cases. *Precision* is the ratio of correctly solved test cases to solved test cases. *F1* score is the harmonic mean of precision and coverage.

SURVEYOR uses a sophisticated probabilistic model to infer the dominant opinion given the counts of positive and

negative statements. We compare this approach against two simpler methods: majority vote and scaled majority vote. *Majority Vote* (MV) decides that a property applies to an entity if the number of positive statements exceeds the number of negative statements and decides that the property does not apply if the number of negative statements exceeds the number of positive statements. No decision is possible if the two statement counters are equal. *Scaled Majority Vote* (SMV) is similar but scales the number of negative statements by the average ratio of positive to negative statements. As we will show this provides a gross adjustment of the inherent bias against negative statements.

As we discuss in Section 8, we are aware of no existing work on associating subjective properties to entities that can work on a Web scale without any supervision. The closest related work is the WebChild knowledge base [22] that associates entities with adjectives. The goal of WebChild is to associate an entity with all applicable adjectives like shape, color, and taste. WebChild does not explicitly model subjectivity and does not include negations of adjectives. Therefore, these comparisons are not entirely fair. In our comparison, we treat the absence of a property for an entity in WebChild as a negative assertion. Therefore, the only reason for loss of coverage for WebChild is that an entity is not contained in the knowledge base.

We first compare these four methods on aggregate performance over all 500 test cases and later present the results against varying levels of agreement among AMT workers. Table 5 shows that SURVEYOR is significantly better than the other methods according to all criteria. MV has low coverage indicative of the many cases with equal number of positive and negative statements which is often zero. SMV is able to improve on test cases where the number of negative statements is non-zero, explaining the slightly increased coverage. The coverage of WebChild is similar to the baseline's. The coverage of SURVEYOR is nearly doubled in comparison to the baselines because SURVEYOR employs a type and property specific model that allows inference even for entities that are not mentioned on the Web. SURVEYOR also performs significantly better in terms of precision. MV has the worst precision since it does not account for polarity bias. SMV has better precision but is still limited by the fact that it assumes universal polarity bias, meaning that its scaling factor is not type and property dependent. The precision of WebChild is significantly higher than for the two baselines. WebChild is however targeted at commonsense properties and not at subjective properties, so it does not explicitly detect negations; for certain property-type combinations (e.g., CUTE ANIMALS) we observed a high number of false positives from WebChild and suspect a connection. SURVEYOR suffers from none of those limitations; it detects negations and its model adapts to the type and property specific polarity bias.

We next compare the different approaches against varying levels of agreement among AMT workers. It seems more important that our system takes the right decisions for entity-property combinations where worker agreement is high. Figure 11 shows how many entity-property combinations satisfy different thresholds on the worker agreement. We observe that there is significant variation in the level of agreement; we have perfect agreement on 180 cases and less than 75% agreement on 100 cases. Figure 12 shows how precision and coverage vary with varying worker agreement. We see that

Table 3: Comparison of different methods for interpreting statement counters

Approach	Coverage	Precision	F1
Majority Vote	0.483	0.29	0.36
Scaled Majority Vote	0.486	0.37	0.42
WebChild	0.477	0.54	0.51
Surveyor	0.966	0.77	0.84

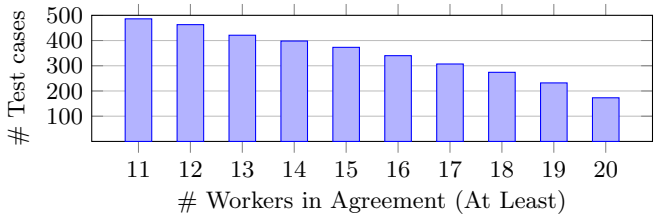


Figure 11: Number of test cases with AMT worker agreement above threshold

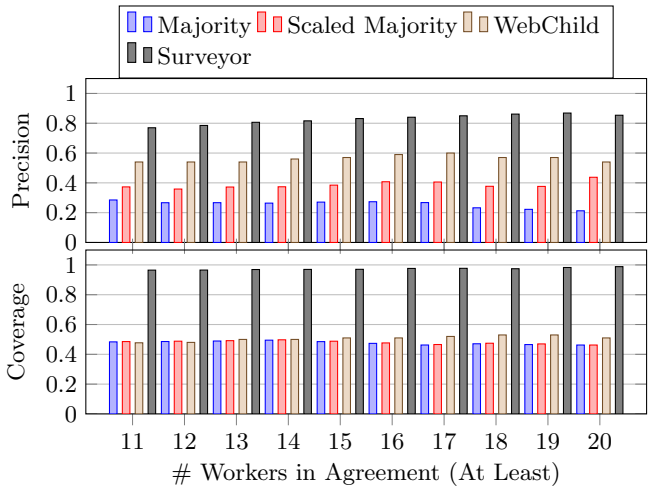


Figure 12: Precision and coverage for test cases with agreement above threshold

the precision of SURVEYOR increases with the worker agreement. It increases from 77% when considering all test cases to 87% when at most one worker disagreed. Figure 11 shows that almost half of all test cases fall into this category. MV cannot benefit from growing worker agreement while the precision of SMV increases from 37% to 44%. The effect is inconclusive for WebChild. Altogether, SURVEYOR performs better for test cases with high worker agreement which are at the same time the test cases that matter most.

We also experimented with randomly selected entities and properties which leads to similar precision values for all approaches while the coverage gap between SURVEYOR and the baselines increases. Details are given in Appendix D.

7.5 Discussion

The results show that our problem scope is reasonable: even if we consider subjective properties, the average agree-

ment of 17 out of 20 for our test cases shows that a dominant opinion exists for many entity-property combinations. SURVEYOR outperforms competing approaches significantly in terms of precision and coverage. We believe that this is mainly due to the following properties of our approach. First, we distinguish between affirmative and negative statements which is important for controversial properties. Second, we do not treat different entities independently from each other but treat all entities of the same type together for each property. This increases coverage since it sometimes allows inferences for entities that are rarely mentioned (this is the case for most entities as shown in Section 7.2). Also, it improves precision since it allows to take type and property specific bias influencing the number of positive and negative statements into account.

8. RELATED WORK

We discuss the most closely related work in this section while we discuss additional publications in Appendix E.

Our work belongs to a broad family of approaches that parse free text to gain information about entities [2, 3, 4, 5, 22]. WebChild [22] associates nouns with adjectives over fine-grained relations such as has-shape or has-taste. The focus is on finding commonsense associations (e.g., carrot has-color orange) that are expected to be noncontroversial (e.g., we do not expect statements claiming that carrots are not orange). Therefore, WebChild does not search for statements claiming that a property does not apply to an entity and has no mechanisms to aggregate conflicting opinions. We compare our system against WebChild in Section 7. Chakrabarti et al. [4] and Cheng et al. [5] associate properties (*tags* in their terminology) with entities if the co-occurrence frequency (i.e., the number of documents in which property and entity appear close to each other) is statistically significant. Chakrabarti et al. [3] and Agrawal et al. [2] associate URLs with entities and rank entities based on the number of associated URLs that a search engine returns when querying for product features. The focus of these papers is less controversial properties like product features. They do not distinguish positive from negative statements, and do not deal with bias in mentions. As shown in Section 7, both of these are essential for subjective properties.

Sophisticated probabilistic models have been developed in the context of *Information Extraction* [8] to consolidate conflicting fact extractions (e.g., [6, 7, 20]). Those models are unsuitable for the scenario that we consider because their purpose is to explain the behavior of extraction algorithms in order to calculate a confidence from the number of consistent extractions. The purpose of our model is to explain the behavior of users in order to infer the majority opinion from the potentially non-representative set of statements that we collect. This is why our model integrates parameters modeling subjectivity and bias between positive and negative statements. This is what enables us to make inference even from the fact that we do not harvest any statements about specific entities. The models that are used in information extraction do not model subjectivity (assuming that inconsistent extractions are always due to mistakes of the extraction algorithm) and all inference is based on the fact that information was extracted (but not on the fact that no information was extracted).

The OpenEval system [19] is similar to our system in that it tries to connect entities to predicates. It differs however from our system since it uses supervised learning and would require us to specify by hand positive and negative samples for each property-type combination; this approach does not scale to large numbers of types and properties. Also, OpenEval uses Google queries to collect statements and is not able to mine a large number of opinions for a given entity-property combination as it only examines the first few result sites for each query. OpenEval is designed for scenarios in which the ground truth can be collected from a few reliable sources and it is not necessary to mine many conflicting opinions in order to obtain a reliable estimate of the majority opinion.

Opinion Mining and Sentiment Analysis [24, 23, 15] techniques extract and aggregate sentiments from text snippets; the goal is to detect expressions of positive or negative sentiments. One of the main challenges in this field is how to associate text snippets with sentiments, which are often expressed in subtle ways. Corresponding approaches classify text snippets on n-grams that are known to be associated with positive or negative sentiments. Such n-grams with associated polarity can be manually entered [21], learned from labeled reviews [16, 15], obtained from synonym and antonym relationships in WordNet[2], or seeded from a manually created list [11]. Our terminology is similar to the one used in sentiment analysis but the semantic differs: positive and negative statements in our case are not associated with positive or negative sentiments. In our scenario, statements connect entities to properties and the entities and properties are explicitly mentioned; there is no need to interpret subtle nuances. Unlike most approaches in sentiment analysis, our method does not require any labeled training samples nor dictionaries; instead, we rely on a fixed set of extraction patterns that are specific to our scenario. Opinion mining and sentiment analysis systems might include a final aggregation stage but this simply involves counting the number of sentences (or reviews) in which positive or negative sentiments are expressed [11, 13, 25] or calculating a rating average [15]. In contrast, our focus is on a intelligent aggregation step that adjusts for user bias and thus goes beyond counting and averaging.

9. CONCLUSION AND OUTLOOK

Users often use subjective properties to restrict the scope of their Web queries. If search engine providers want to be able to respond to such queries with structured information then they must learn how to associate entities with subjective properties. SURVEYOR represents our first step into this direction. Our experimental results justify our problem scope as well as our overall approach.

In future work, we plan to use SURVEYOR to connect subjective properties to objective properties. We could for instance try to find a lower bound on the population count of a city starting from which an average user would call that city BIG. Inferring and exploiting such relationships should allow to improve precision and coverage for the subset of subjective properties for which such correlations can be found.

10. ACKNOWLEDGEMENTS

We thank Niket Tandon for his support during the comparison of SURVEYOR against the WebChild system.

[2]http://wordnet.princeton.edu

11. REFERENCES

[1] Special issue: nonresponse bias in household surveys. *Public Opinion Quarterly*, 70(5), 2006.

[2] S. Agrawal and K. Chakrabarti. Query portals: dynamically generating portals for entity-oriented web queries. *SIGMOD*, pages 1171–1174, 2010.

[3] K. Chakrabarti. Ranking objects by exploiting relationships: computing top-k over aggregation. *SIGMOD*, pages 371–382, 2006.

[4] K. Chakrabarti and S. Chaudhuri. Entitytagger: automatically tagging entities with descriptive phrases. *WWW*, pages 3–4, 2011.

[5] T. Cheng, K. Chakrabarti, S. Chaudhuri, V. Narasayya, and M. Syamala. Data services for e-tailers leveraging web search engine assets. In *ICDE*, pages 1153–1164, 2013.

[6] A. Culotta and A. McCallum. Confidence estimation for information extraction. *HLT-NAACL*, pages 109–112, 2004.

[7] D. Downey, O. Etzioni, and S. Soderland. A probabilistic model of redundancy in information extraction. In *IJCAI*, pages 1034–1041, 2005.

[8] O. Etzioni, M. Banko, S. Soderland, and D. S. Weld. Open information extraction from the web. *Communications of the ACM*, 51(12):68–74, 2008.

[9] R. Fisher. On an absolute criterion for fitting frequency curves. *Messenger of Mathematics*, 41:155–160, 1912.

[10] C. Giuliano. Fine-grained classification of named entities exploiting latent semantic kernels. In *CoNLL*, pages 201–209, 2009.

[11] M. Hu and B. Liu. Mining and summarizing customer reviews. *KDD*, pages 168–177, 2004.

[12] X. Ling and D. Weld. Fine-grained entity recognition. In *AAAI*, pages 94–100, 2012.

[13] B. Liu, M. Hu, and J. Cheng. Opinion observer: analyzing and comparing opinions on the web. In *WWW*, pages 342–351, 2005.

[14] D. McDonald. On the poisson approximation of the multinomial distribution. *Canadian Journal of Statistics*, 8(1):115–118, 1980.

[15] S. Moghaddam and M. Ester. Opinion digger. In *CIKM*, pages 18–25, 2010.

[16] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up? Sentiment classification using machine learning techniques. In *Empirical methods in natural language processing*, pages 79–86, 2002.

[17] A. Popescu, B. Nguyen, and O. Etzioni. OPINE: Extracting product features and opinions from reviews. In *HLT/EMNLP on interactive demonstrations*, pages 32–33, 2005.

[18] B. Roos. On the rate of multivariate Poisson convergence. *Journal of Multivariate Analysis*, 69(1):120–134, 1999.

[19] M. Samadi and M. Blum. OpenEval : Web information query evaluation. In *AAAI*, pages 1163–1169, 2013.

[20] M. Skounakis and M. Craven. Evidence combination in biomedical natural-language processing. *BIOKDD*, pages 2–9, 2003.

[21] M. Taboada, J. Brooke, and M. Tofiloski. Lexicon-based methods for sentiment analysis. *Computational linguistics*, 37(2):267–307, 2011.

[22] N. Tandon, G. de Melo, F. Suchanek, and G. Weikum. WebChild: harvesting and organizing commonsense knowledge from the Web. In *WSDM*, pages 523–532, 2014.

[23] M. Tsytsarau, S. Amer-Yahia, and T. Palpanas. Efficient sentiment correlation for large-scale demographics. *SIGMOD*, pages 253–264, 2013.

[24] M. Tsytsarau and T. Palpanas. Survey on mining subjective data on the web. *KDD*, 24(3):478–514, 2011.

[25] L. Zhuang, F. Jing, and X.-Y. Zhu. Movie review mining and summarization. In *CIKM*, pages 43–50, 2006.

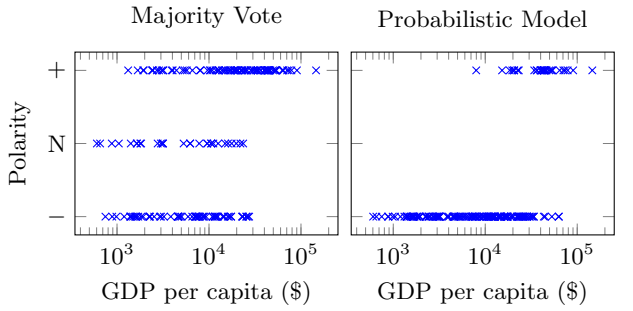
APPENDIX

A. EXTENDED EMPIRICAL STUDY

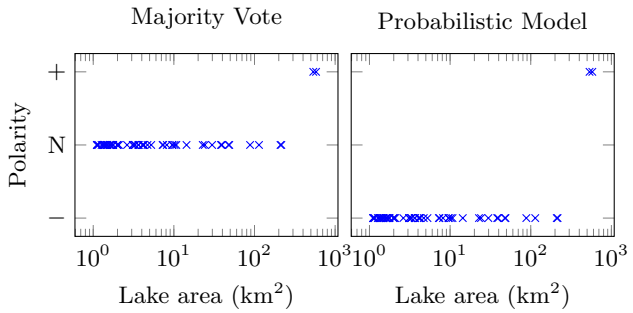
We describe the results for additional property-type combinations that we tried during our empirical study. In addition to the property-type combination BIG CITY, we tried the combinations WEALTHY COUNTRY, BIG LAKE, and HIGH MOUNTAIN. The experimental setup is the same as in Section 2: for each subjective property, we determine a numerical property that it should correlate with. Whether a country is considered wealthy should correlate with its GDP per capita. Whether a lake is considered big or not should correlate with its area and whether a mountain is considered high should correlate with its height.

For each of the three property-type combinations, we identified an HTML table on the Web that contains entities of the corresponding type and specifies the required numerical property for them. We selected a table containing countries with their GDP per capita as estimated by the IMF for 2013, a table containing lakes in Switzerland with their associated area in square kilometers, and a table containing mountains on the British Islands with their relative height in meters. We issued Google queries for all entities in the corresponding tables as described in Section 2, thereby counting affirmative and negative statements on the Web. Then we use the majority vote and the probabilistic model to estimate for each entity the polarity from its statement counts.

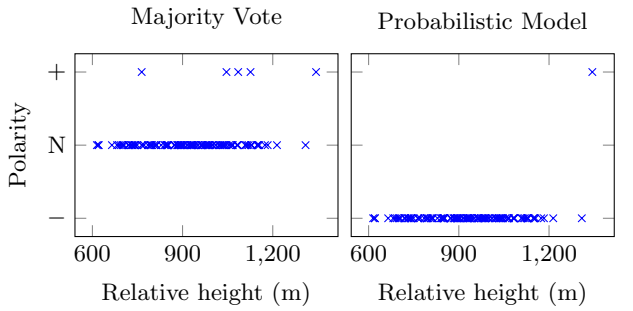
Figure 13 compares the polarity results in terms of how well they correlate with the numerical properties that we specified before. For all three scenarios, the correlation is significantly better for the probabilistic model. In addition, the probabilistic model is even able to classify entities for which no statements were collected. The latter problem concerns in particular the scenarios with lakes and mountains. Those scenarios are representative for the bigger part of property-type combinations that SURVEYOR considers; as our knowledge base is large, it contains many entities for which no statements can be collected on the Web (see statistics in Section 7.2). The SURVEYOR system can cope with sparseness since it considers sets of entities of the same type together; the majority vote approach considers each entity separately and cannot deal well with sparseness. Note that we use a qualitative evaluation approach here, while we use a quantitative approach to compare the probabilistic model against majority vote and other baselines in the experimental section.



(a) Wealthy countries



(b) Big lakes in Switzerland



(c) High mountains on the British Islands

Figure 13: Result comparisons of majority vote (left side) and probabilistic model (right side).

B. EVOLUTION OF PATTERNS

We provide details on the different extraction patterns that we tried out before opting for the version that is presented in Section 4. Table 4 shows the different versions that we developed and tried out consecutively. The modifiers column describes the types of connections between adjectives and nouns that we considered for extraction; amod abbreviates adjectival modifier and captures cases such as "the cute cat" while acomp abbreviates accompanying modifier and captures cases such as "the cat is cute". The verbs column describes the verbs we considered for connecting adjectives to nouns; the class of copula verbs comprises all verbs that are commonly used to connect the subject of a sentence to a predicate. The other alternative is to be more restrictive and to consider only the verb "to be". The checks column indicates whether we filter out non-intrinsic statements. The statements column describes how many statements were extracted by the corresponding version.

Table 4: Comparison of different pattern versions

Vers.	Modifiers	Verbs	Check	Statements
1	amod	copula	no	1321194344
2	amod+acomp	copula	no	1779253966
3	acomp	to be	yes	98574972
4	amod+acomp	to be	yes	922299774

Our first version did not filter out non-intrinsic statements and used only the adjectival modifier pattern. After adding the accompanying modifier pattern for the second version, the number of extracted statements increased by over 30%. While the number of extractions was satisfactory, the quality of the extracted statements was not sufficient as we verified by studying example extractions. Our goal for the third version was to improve extraction quality; we therefore added the intrinsicness tests that are described in Section 4 and restricted ourselves to the accompanying modifier pattern and the verb "to be". This increased extraction quality indeed but at the same time decreased the number of extractions by more than one order of magnitude. For the final version, we used the adjectival modifier and accompanying modifier patterns together again but left the intrinsicness checks in place. The quality of the extractions now seemed significantly better than in the first versions while the number of extracted statements was still relatively close to the maximum. We used the fourth version for the experiments as it seemed to offer the best trade-off between precision and recall.

We also report extraction times for the different versions of the extraction patterns. Using version 1, extraction took around 2 hours and 40 minutes on a cluster with 1000 nodes. We increased the cluster size for executing version 2; extraction took around 50 minutes on a cluster with 5000 nodes. Version 3 took around 48 minutes with 5000 nodes. For our final version, extraction time was around one hour. Note that run time was not our primary criterion for selecting between extraction patterns.

C. ANALYSIS DETAILS

We provide more details on the analysis that we sketched in Section 6, leading to our formulas for calculating optimal parameter values. More precisely, we show how we obtain function $Q'_k(\Theta)$ which becomes maximal for the same parameter values as $Q_k(\Theta)$. The formula for Q_k is:

$$\sum_{i=1}^{m} \sum_{d_i \in \{+,-\}} \Big[\log(\Pr(D_i = d_i, E_i|\theta)) \Pr(D_i = d_i|\theta_{k-1}, E_i) \Big]$$

The factor $\Pr(D_i = d_i|\theta_{k-1}, E_i)$ is a constant in iteration k and we focus on transformations of the factor $\log(\Pr(D_i = d_i, E_i|\theta))$ in the following. The following transformations use the formula of conditional probabilities and basic properties of the logarithm function:

$$\begin{aligned} &\log(\Pr(D_i = d_i, E_i|\theta)) \\ =&\log(\Pr(D_i = d_i|\theta) \cdot \Pr(E_i|D_i = d_i, \theta)) \\ =&\log(\Pr(D_i = d_i|\theta)) + \log(\Pr(E_i|D_i = d_i, \theta)) \end{aligned}$$

The term $\log(\Pr(D_i = d_i|\theta))$ does not directly depend on the parameters since we do not consider the evidence E_i yet (the parameters concern only the connection between the probability for the dominant opinion and the collected evidence). We can therefore neglect the term when maximizing Q_k as it has no influence on the position of the maximum. We focus on the term $\log(\Pr(E_i|D_i = d_i, \theta))$ in the following and only consider the special case $D_i = d_i = +$ (while the analysis for the opposite case is analogue). The following transformations exploit the definition of the evidence $E_i = (C_i^+ = c_i^+ \wedge C_i^- = c_i^-)$ (i.e., the evidence consists of receiving a specific number c_i^+ of positive and a specific number c_i^- of negative statements), the assumed independence between the counts of positive and negative statements, the assumption that the number of evidence statements is distributed according to the Poisson distribution, and basic properties of the logarithm function:

$$\begin{aligned}
&\log(\Pr(E_i|D_i = +, \theta)) \\
=&\log(\Pr(C_i^+ = c_i^+|D_i = +, \theta) \cdot \Pr(C_i^- = c_i^-|D_i = +, \theta)) \\
=&\log(\text{Pois}(c_i^+; \lambda_+^+) \cdot \text{Pois}(c_i^-; \lambda_+^-)) \\
=&\log(\text{Pois}(c_i^+; \lambda_+^+)) + \log(\text{Pois}(c_i^-; \lambda_+^-))
\end{aligned}$$

We focus on the term $\log(\text{Pois}(c_i^+; \lambda_+^+))$ in the following while the analysis of the term $\log(\text{Pois}(c_i^-; \lambda_+^-))$ is analogue. We use the definition of the Poisson distribution and the logarithm rules:

$$\begin{aligned}
&\log(\text{Pois}(c_i^+; \lambda_+^+)) \\
=&\log(\frac{(\lambda_+^+)^{c_i^+} e^{-\lambda_+^+}}{c_i^+!}) \\
=&\log((\lambda_+^+)^{c_i^+} e^{-\lambda_+^+}) - \log(c_i^+!)
\end{aligned}$$

The term $\log(c_i^+!)$ does not depend on θ and we can neglect it when maximizing Q_k in θ. We transform the remaining term $\log((\lambda_+^+)^{c_i^+} e^{-\lambda_+^+})$ using our definition of the Poisson parameter λ_+^+:

$$\begin{aligned}
&\log((\lambda_+^+)^{c_i^+} e^{-\lambda_+^+}) \\
=&c_i^+ \cdot \log(\lambda_+^+) - \lambda_+^+ \\
=&c_i^+ \cdot \log(np_A p_S^+) - np_A p_S^+
\end{aligned}$$

We can apply analogue transformations to all terms in Q_k to obtain the formula for Q'_k that was specified in Section 6.

D. ADDITIONAL EXPERIMENTS

For the experiments in Section 7, we selected entity types and properties that frequently appear in queries. This reflects our main application scenario of interpreting queries. In this section, we provide additional results for types and properties that are sampled randomly from our large result set. We sampled 803 property-type combinations and for each combination we randomly sampled seven entities. This results in a total of over 5500 randomly sampled test cases.

We use those test cases to compare the performance of the same four approaches that we already evaluated in the experimental section. We used all test cases to evaluate coverage for the four approaches, as coverage can be calculated automatically: for scaled and naive majority vote we just verify whether the scaled or raw number of affirmative and negative statements are equal, for WebChild we check whether the entity that the test case refers to is included in the knowledge base.

Table 5: Comparison for random sample of property-type combinations

Approach	Coverage	Precision	F1
Majority Vote	0.0766	0.333	0.125
Scaled Majority Vote	0.0773	0.417	0.130
WebChild	0.173	0.615	0.270
Surveyor	0.999	0.784	0.879

Precision cannot be checked automatically but requires ground truth data to compare the output of the four approaches against. We therefore selected 80 property-type combinations and generated ground truth data for one randomly sampled entity per combination. We excluded six clearly offensive combinations such as "obnoxious ethnicity". In contrast to Section 7, we did not use AMT to generate the ground truth data. The reason is the following: our knowledge base is very large and when selecting random entities, we obtain in most cases entities that are very specific and not known to the general public. Our random sample of test cases includes for instance the Latin name of a disease ("Hiatal hernia"), a Portuguese artist named "Maria Lusitano", and the car model "Ford Cougar". Deciding for instance whether hiatal hernia should be counted as a MAJOR DISEASE required us to execute an Internet search and to read through corresponding material. We therefore do not expect to obtain meaningful results from AMT as this platform is rather targeted at simple and quickly executable tasks. The types and properties that we selected in Section 7 (e.g., animals) are more appropriate for an evaluation using AMT since those entities should be known to the general public.

Note that the aforementioned entities might still be useful to answer specific queries (e.g., a query asking for major diseases with specific properties that is issued by a person with a medical background).

Table 5 shows the results: our system is the only one which can slightly improve its F1 score comparing with the results in Section 7. For the three other approaches, the drop in F1 score is significant. In particular, there is a drop in coverage since the randomly sampled entities are less frequently mentioned on the Web, while the values for precision are comparable to the ones in Section 7. While the results in Table 5 are very favorable for our system, we still believe that the results reported in Section 7 are more relevant since they refer to entities that we expect to be more common in queries than the ones we randomly sampled. Our goal was however to show that our selection of test cases did not penalize the baseline approaches.

E. ADDITIONAL RELATED WORK

We pick one popular probabilistic model from *Information Extraction*, the one proposed by Downey et al. [7], as example to show why such models are ill-suited for our task. The model by Downey et al. models each extractor by a

probabilistic "urn", fact extractions (e.g., the fact that New York is a city) are modeled as draws from the urns. The model calculates a confidence that specific facts are correct, based on the number of times the corresponding fact was extracted and the total number of draws. This model is not suitable for our scenario for several reasons. First, it cannot infer anything for the common case that no extractions were made for specific entity-property pairs; our model can give meaning to that case since it connects the likelihood that users express their opinion to their opinion. Second, the model by Downey et al. does not explicitly take into account mutual exclusion constraints between extracted facts. To map our scenario onto their model, we need to conceptually introduce for each property one extractor for positive statements and one extractor for negative statements. Then the latter model might for a given city calculate a probability for that city being big according to the dominant opinion and a probability for that city being not big such that those two probabilities do not sum up to one. The model that we introduce in this paper takes such exclusions into account.

Entity Classification [10, 12] is a sub-field of information extraction; the goal is to infer the type of entities mentioned in free text, using the surrounding text for the classification. Such techniques are a prerequisite for our approach since we group entities according to their type in order to learn type-specific content generation patterns. At the same time, the problem solved by SURVEYOR can be seen as a form of entity classification as well since for each property we classify entities into those to which the property applies and those where it does not apply. Our scenario poses however specific challenges since we consider properties that are controversially discussed; our prior discussion contrasting the probabilistic models used in information extraction from our model applies as well.

Feature Mining is often used as pre-processing step for opinion mining [11, 17, 15]; Hu and Liu [11] mine for instance product features (e.g., "picture quality" as feature of "camera") before extracting ratings for those features in a second step (we already discussed approaches for extracting ratings in the previous paragraph). Features are associated with entity types while subjective properties are associated with entities (e.g., the feature "picture quality" applies to all cameras, independently of whether they have good or bad picture quality, while the property "cute" only applies to a subset of animals but not to all of them). So the problem model of feature mining does not map to the problem model of subjective property mining. Also, features are usually frequent nouns [15] while subjective properties are usually adjectives and adverbs and associating properties with entities based on occurrence frequency alone creates problems since it neglects negation.

In *Classical Surveys*, participants are explicitly asked for their opinion. The bias introduced by the fact that some participants refuse to participate in a survey is known as non-response bias [1]. This case is considered rather rare and research usually focuses on bounding the effect of this bias or trying to avoid it by appropriate survey design. Our scenario differs from a traditional survey since we do not solicit participation but rather collect opinions that users decided to post themselves. This makes the "non-response" bias an important fact in our case since we have no control over the selection of participants. On the other side, our survey is simplistic comparing with traditional surveys, as we only have two options for each entity-property combination. This gives us the possibility to sometimes even infer something from the fact that we find "no participants".