Was It Worth It? Summarizing and Navigating User Reviews with Natural Language Methods

Eric Gilbert and Karrie Karahalios
University of Illinois at Urbana-Champaign
[egilber2, kkarahal]@cs.uiuc.edu

ABSTRACT
Popular products often attract an astonishing number of user reviews. Sifting through them by the thousands can be quite a headache, one that has inspired solutions like unique phrase identification (e.g., Yelp) and helpfulness ratings (e.g., Amazon). While clearly useful, these techniques only capitalize on a small subset of the reviews, and cannot answer questions like, "do these people care about the same things I care about?" *Talking Points* is our solution to these problems. It employs natural language methods to summarize thousands of user reviews into a navigable, browser-based interface. In addition to describing our novel algorithm for feature extraction and sentiment classification, this paper presents the results of an exploratory user study of *Talking Points.* Our study suggests that users explore reviews far longer with *Talking Points* than with traditional methods. More surprisingly, in randomized sessions users seemed persuaded to choose those products augmented with *Talking Points*.

Author Keywords
Social media, user reviews, summarization, nlp

ACM Classification Keywords
H5.3. Group and Organization Interfaces; Asynchronous interaction; Web-based interaction.

INTRODUCTION
A popular product can spark thousands of people to speak their minds. For instance, the television series *Firefly* has more than 2,500 Amazon customer reviews; 93% of them are five-stars. How can users absorb 2,500 reviews? Yes, almost everyone *loves* it, but why? And do they care about the same things I care about (e.g., "I don't care much for special effects, but character development is essential")? In parallel, how can sites efficiently use thousands of reviews to help a new user?

In this paper, we explore these questions and present a solution: *Talking Points*, an interface that uses natural language methods to summarize user reviews in a navigable interface. In contrast with related work primarily from machine learning venues [1, 2, 8, 9, 13, 14, 16], we designed *Talking Points* to be "user-grade." Design is a priority. Our novel algorithm aims for very high confidence because users explore its results. Instead of a precision-recall experiment, we present an evaluation of *Talking Points* in the hands of people deciding on a product, their own compensation for our study.

In our study, we learned two particularly interesting things. First, users spent over 50% more time exploring reviews on product pages that included *Talking Points*. More surprisingly, although we randomized the appearance of *Talking Points,* seven of our eight participants picked products from *Talking Point*s pages. By chance alone, we would expect this to happen less than 4% of the time. From this and other data collected during our study, we argue that *Talking Points* is both a *persuasive* and *useful* way to navigate an ever-growing number of user reviews.

DESIGN & INTERACTION
The problem of guiding users through hundreds or thousands of reviews has inspired useful solutions like Yelp's unique phrase identification and Amazon's helpfulness ratings. However, unique phrase identification assumes that people no longer care about common words, like a restaurant's "food." Helpfulness ratings, an elegant idea, require work from users and quickly create a stampede toward one or two reviews. In *Talking Points,* we take an orthogonal design approach: across all reviews, we visualize the things users talk about most along with the positive and negative ways users describe those things. Figures 1 and 2 show *Talking Points* embedded on an Amazon page. We designed the interface to fit into sites full of other, more important content. We realize that users will not come to a site to use *Talking Points*; they come to buy a product (e.g., Amazon), learn about a restaurant (e.g., Yelp) or research a trip (e.g., TripAdvisor). For that reason, we designed our interface to live in a small space and to distract as little as possible.

In its first-loaded form, *Talking Points* presents a user with a simple list of features, grouped by the overall tone of the discussion about them. Features discussed positively twice as often as negatively are grouped together by a red heart. Features twice as negative as positive are signified by a black broken heart. An empty heart groups the ones in between. Users may click on a feature to explore the words used to describe it. When a user clicks on one of these ad-

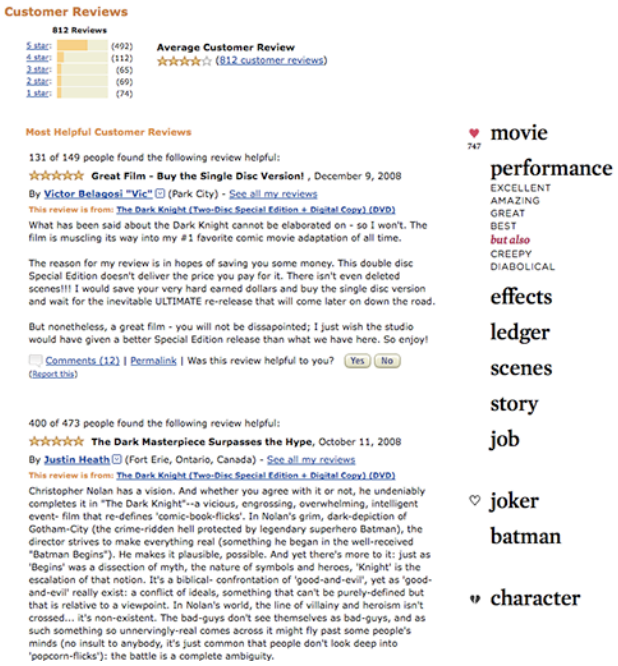

Figure 1. *Talking Points* summarizing *The Dark Knight*, as reviewed by Amazon.com's customers. Users explore reviews through automatically extracted features and the words used to describe them. The overall tone of a feature's discussion places it into one of three categories, each depicted by a heart.

jectives or adverbs, they see excerpts of the reviews from which we collected the words (see accompanying video). Instead of distinguishing a feature's positive words from its negative ones by size, typeface or color, we apply a relatively unique approach: animation. Figure 4 illustrates our design. Infrequently and in small doses, words take on behaviors to embody their polarity. Positive words bounce; negative words fall away. This happens rarely, less than 10% of the time, and only during interaction with *Talking Points*. The rest of the time, it remains easy to scan.

Within each heart-group, *Talking Points* presents features ordered by number of mentions. The adjectives and adverbs used to describe them, however, are ordered by a slightly more complex algorithm. We rank them using TF-IDF [6] modified to incorporate the strength of a word (provided by SentiWordNet [4]).

IMPLEMENTATION

The *Talking Points* interface is built in Flash. Behind the scenes, product reviews undergo a five-stage algorithm before converging on Figure 2: 1) crawling & cleaning; 2) a full statistical parse; 3) feature & adjective/adverb extraction; 4) feature stemming; and, 5) processing by four sentiment models. Figure 3 presents the algorithm compactly.

First, a script crawls and cleans the reviews. Next, we apply the statistical parser from [7] to each sentence, generating a full parse tree, a dependency parse and the part-of-speech (POS) tag for each word. A full parse allows us to understand reviews as deeply as the state of the art allows. However, this comes at price: parsing is computationally expen-

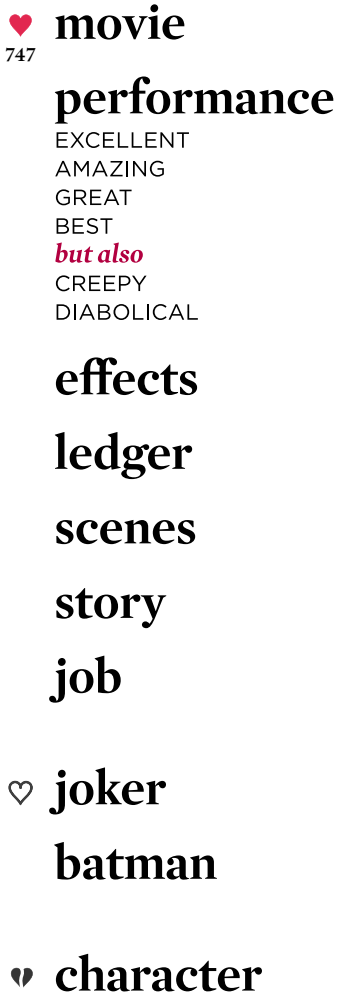

Figure 2. An up-close view of the interface in Fig. 1. The user expands *performance* to uncover the positive and negative words used to describe it. The word *diabolical*, only mentioned a handful of times, appears because of the force behind the word. Users click adjectives to see excerpts of the reviews in which it appears.

sive. For example, parsing a product with 431 reviews (an unremarkable number) took 118 minutes on a 1.8GHz dual-core machine with 2GB of RAM. While computationally expensive, two features make it tractable: the process need only occur once for existing reviews; and, the reviews are data-parallel. In other words, if Amazon or Yelp wished to apply this method to all its reviews, it need only find a large cluster and process them once. Since sentences are independent, the computation is easily parallelized.

Next, we scan the dependency parse tree and POS tags for pairs in which an adjective or adverb modifies a noun, noun phrase or verb (*nsubj, amod* and *acomp* links in the language of [3]). This scan is written in Perl and runs quickly, often in less 15 seconds on the machine described above. Note that our algorithm does not simply scan sentences for adjectives and nouns in sequence. Phrases like "imaginative, exciting, albeit poorly executed production" would confuse this simple strategy. During the scan, we also make

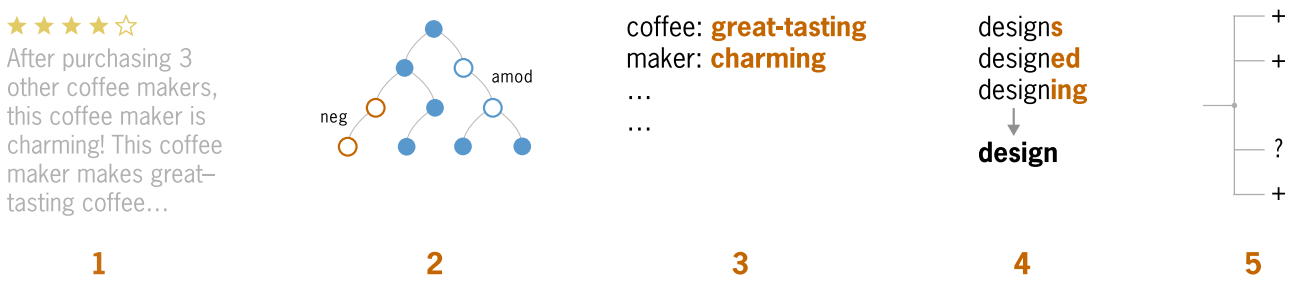

Figure 3. Our algorithm extracts features and the positive & negative words used to describe them. 1) Reviews are crawled and scrubbed. 2) For each sentence, we compute the full parse tree, a dependency parse and the POS tags. 3) The dependency parse is scanned for feature-adjective/adverb pairs, accounting for negations. 4) Features coalesce through stemming. 5) A mixture of four sentiment models decides the polarity of an adjective or adverb, via the ensemble inequality below.

note of negations in the tree and invert the meaning of the affected child nodes. For instance, we do not want to call the phrase "isn't nearly the remarkable battery life I thought I was buying" an endorsement. Once all the features and adjective/adverbs are gathered, we apply a stemmer [12] to merge isomorphic features. We explored going one step further and combining synonymous features using WordNet [5]. Is a "carafe" really that different from a "pot," after all? However, experimenting with this technique produced discouraging results. In the end, we decided to let the user explore the subtle differences between similar features.

Finally, we decide on the subjective polarity of the adjectives and adverbs; that is, whether (and how strongly) the adjectives and adverbs are positive or negative. Since we let users navigate reviews concretely via words, we must be confident about the judgements. Yet, most sentiment models have significant weaknesses (limited vocabularies, domain specificity, etc.) that would produce unacceptable mistakes. To overcome this problem, we apply a mixture of four sentiment models, relying on their (relative) orthogonality to work in our favor: LIWC [11], the Rotten Tomatoes corpus [10], the Congressional speeches corpus [15] and SentiWordNet [4]. If an adjective or adverb's stem appears on LIWC's positive or negative list, our algorithm trusts its decision. LIWC is a hand-compiled list covering many common words. However, most words don't appear on its lists. To classify these words, we use an ensemble of probabilistic classifiers. Specifically, the following *ensemble inequality* must hold for a word w to be considered positive:

$$I[\frac{RTPos(w)}{RTNeg(w)} > 2] + I[\frac{CPos(w)}{CNeg(w)} > 2] +$$
$$I[SWNPosVotes(w) > 0] + I[SWNMax(w) > 0.25] +$$
$$I[SWNSum(w) > 0.25] + I[SWNFirst(w) > 0.25] \geq 4$$

Negative words have a parallel inequality. $I[x]$ is 1 when the expression x holds and 0 otherwise. The first term is 1 when a word appears in positive Rotten Tomatoes reviews twice as often as in negative ones. The second term expresses the same idea for Congressional speeches. *SWNPosVotes* counts to the number of senses of w which SentiWordNet thinks are positive. *SWNMax* and *SWNSum* refer to the maximum and sum over all senses of w in SentiWordNet. These three measures smooth over the many senses a word can take, freeing our algorithm from resolving collocation and word sense disambiguation problems. Finally, we give special priority to a word's most common sense: *SWNFirst* is the score of the most common sense of w. Admittedly, the ensemble inequality is the result of educated guesswork and trial and error. There are few alternatives, and our first priority is a usable system. The ensemble inequality is also rather conservative for this reason: a word gets a positive or negative label only when at least four sources agree.

EVALUATION

We performed an exploratory lab study to evaluate *Talking Points*. Eight undergraduate and graduate students were recruited to use *Talking Points* to select a DVD from Amazon as their compensation for the study. Participants spent roughly 30 minutes using a Firefox browser instrumented with Greasemonkey to decide on a bestselling Amazon DVD. (However, the study ended as soon as participants made a choice.) We designed the study this way to incentivize participants' behavior. We held price roughly constant (~$17) and deleted every price from the page to counteract the effect of small price differences between products.

Our script randomly modified half of the DVD pages to include *Talking Points*. Figure 1 shows exactly what participants saw. The other half, the control pages, were normal Amazon pages. In addition to gathering qualitative feedback, we looked to answer the following research questions. Do participants deliberate more with *Talking Points*? Does *Talking Points* affect participants' confidence about their decisions? Their satisfaction?

Results

When asked about *Talking Points* pages (the experimental condition), users reported that the interface was quite useful for making decisions, med. = 6, μ = 5.15, σ = 0.95. (All numeric questionnaire data falls on a 7-point Likert scale.) One participant said that *Taking Points* "does something I already do in my head … I'll often read the first ten reviews or so to find the commonalities." Another participant hated "when reviews give away the story," and felt that *Talking Points* provided a clear summary without divulging too much. However, two participants felt that the technique would provide more insight into something more tangible than movies. Participant opinions on the design elements of *Talking Points* agreed with the overall assessment, with users most appreciating its ability to summarize hundreds or thousands of reviews (med. = 6.5, μ = 6.25, σ = 0.87) and

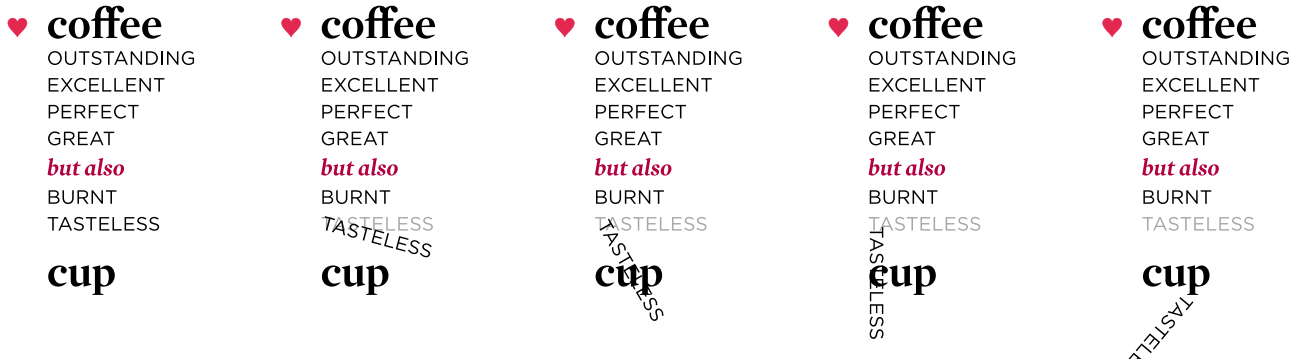

Figure 4. Our technique applied to a popular coffee maker on Amazon.com. Instead of rendering positive & negative words in different fonts, colors or sizes, we apply a unique design choice to remind users of their polarity: animation. Infrequently and in small doses, positive and negative words take on behaviors. Here, a negative words swings, bangs onto the vertical axis and then falls away. Positive words, on the other hand, bounce.

finding the most room for improvement with feature extraction (med. = 6, $\mu = 5.63$, $\sigma = 1.06$). Some participants mentioned feature extraction in follow-up interviews too, noting that features like "job" in Fig. 2 (e.g., "What a great job!") should probably be eliminated. By and large, however, participants were tolerant of the small number of language mistakes, agreeing that the benefits outweighed them.

Participants felt marginally more confident about their decisions in the *Talking Points* condition ($\mu = 5.5$) than the control condition ($\mu = 5.13$), but this difference was not significant, $t(76) = 1.13$, $p = 0.26$. However, participants spent far longer deliberating on *Talking Points* pages (randomized across sessions) than on control pages, $t(73) = 2.09$, $p = 0.04$. On average, participants spent 98 seconds deliberating about a control DVD, but 152 seconds considering a *Talking Points* DVD. This difference is starker when considering medians: 119 sec. (*Talking Points)* vs. 69 sec. (control).

We had hoped to assess the difference in satisfaction between those participants who selected *Talking Points* DVDs and those who selected control DVDs, expecting a roughly equal split. But a curious thing happened. Seven of our eight participants chose *Talking Points* DVDs. If chance were acting alone, we would expect this to happen less than 4% of the time, $p = 0.035$ (binomial test). In fact, in an unprompted complaint, the one participant who selected a non-*Talking Points* DVD said, "if the visualization had been available for even just one of the three or so movies that I had heard good things about, I would have chosen it."

In conclusion, when taken together, these quantitative and qualitative results suggest that users find *Talking Points* both useful and persuasive. Moreover, we believe our approach demonstrates the successful application of messy, probabilistic tools to a concrete user need.

REFERENCES

1. Archak, N., Ghose, A., et al. Show me the money!: deriving the pricing power of product features by mining consumer reviews. *Proc. KDD,* 2007. 56–65.
2. Dave, K., Lawrence, S., et al. Mining the peanut gallery: opinion extraction and semantic classification of product reviews. *Proc. WWW '03,* 2003. 519–528.
3. de Marneffe, M. C., Maccartney, B., et al. Generating Typed Dependency Parses from Phrase Structure Parses.. *Proc. LREC,* 2006.
4. Esuli, A. and Sebastiani, F. SentiWordNet: A Publicly Available Lexical Resource for Opinion Mining. *Proc. LREC,* 2006.
5. Fellbaum, C. *WordNet: An Electronic Lexical Database.* The MIT Press, 1998.
6. Frakes, W. B. and Baeza-Yates, R. *Information Retrieval: Data Structures and Algorithms.* Prentice Hall, 1992.
7. Klein, D. and Manning, C. D. Accurate unlexicalized parsing. *Proc. ACL,* 2003. 423–430.
8. Liu, B., Hu, M., et al. Opinion observer: analyzing and comparing opinions on the Web. *Proc. WWW,* 2005. 342–351.
9. Pang, B. and Lee, L. *Opinion Mining and Sentiment Analysis.* Now Publishers, 2008.
10. Pang, B., Lee, L., et al. Thumbs up?: sentiment classification using machine learning techniques. *Proc. EMNLP,* 2002. 79–86.
11. Pennebaker, J. W. and Francis, M. E. *Linguistic Inquiry and Word Count.* Lawrence Erlbaum, 1999.
12. Porter, M. F. 1980. An algorithm for suffix stripping. *Program, 14*(3), 130–137.
13. Scaffidi, C., Bierhoff, K., et al. Red Opal: product-feature scoring from reviews. *Proc. EC,* 2007. 182–191.
14. Sun, J., Wang, X., et al. CWS: a comparative web search system. *Proc. WWW,* 2006. 467–476.
15. Thomas, M., Pang, B., et al. Get out the vote: Determining support or opposition from Congressional floor-debate transcripts. *Proc. EMNLP,* 2006. 327–335.
16. Zhuang, L., Jing, F., et al. Movie review mining and summarization. *Proc. CIKM,* 2006. 43–50.