Content Models with Attitude

Christina Sauper, Aria Haghighi, Regina Barzilay
Computer Science and Artificial Intelligence Laboratory
Massachusetts Institute of Technology
csauper@csail.mit.edu, me@aria42.com, regina@csail.mit.edu

Abstract

We present a probabilistic topic model for jointly identifying properties and attributes of social media review snippets. Our model simultaneously learns a set of properties of a product and captures aggregate user sentiments towards these properties. This approach directly enables discovery of highly rated or inconsistent properties of a product. Our model admits an efficient variational mean-field inference algorithm which can be parallelized and run on large snippet collections. We evaluate our model on a large corpus of snippets from Yelp reviews to assess property and attribute prediction. We demonstrate that it outperforms applicable baselines by a considerable margin.

Coherent property cluster

+	The **martinis** were very good.
	The **drinks** - both **wine** and **martinis** - were tasty.
-	The **wine list** was pricey.
	Their **wine selection** is horrible.

Incoherent property cluster

+	The sushi is the **best I've ever had**.
	Best paella **I'd ever had**.
	The fillet was the **best steak we'd ever had**.
	It's the **best soup I've ever had**.

Table 1: Example clusters of restaurant review snippets. The first cluster represents a coherent *property* of the underlying product, namely the *cocktail* property, and assesses distinctions in user sentiment. The latter cluster simply shares a common attribute expression and does not represent snippets discussing the same product property. In this work, we aim to produce the first type of property cluster with correct sentiment labeling.

1 Introduction

Online product reviews have become an increasingly valuable and influential source of information for consumers. Different reviewers may choose to comment on different properties or aspects of a product; therefore their reviews focus on different qualities of the product. Even when they discuss the same properties, their experiences and, subsequently, evaluations of the product can differ dramatically. Thus, information in any single review may not provide a complete and balanced view representative of the product as a whole. To address this need, online retailers often use simple aggregation mechanisms to represent the spectrum of user sentiment. For instance, product pages on Amazon prominently display the distribution of numerical scores across reviews, providing access to reviews at different levels of satisfaction.

The goal of our work is to provide a mechanism for review content aggregation that goes beyond numerical scores. Specifically, we are interested in identifying fine-grained product properties across reviews (e.g., *battery life* for electronics or *pizza* for restaurants) as well as capturing attributes of these properties, namely aggregate user sentiment.

For this task, we assume as input a set of product review snippets (i.e., standalone phrases such as "battery life is the best I've found") rather than complete reviews. There are many techniques for extracting this type of snippet in existing work; we use the Sauper et al. (2010) system.

Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics, pages 350–358,
Portland, Oregon, June 19-24, 2011.

At first glance, this task can be solved using existing methods for review analysis. These methods can effectively extract product properties from individual snippets along with their corresponding sentiment. While the resulting property-attribute pairs form a useful abstraction for cross-review analysis, in practice direct comparison of these pairs is challenging.

Consider, for instance, the two clusters of restaurant review snippets shown in Figure 1. While both clusters have many words in common among their members, only the first describes a coherent property cluster, namely the *cocktail* property. The snippets of the latter cluster do not discuss a single product property, but instead share similar expressions of sentiment. To solve this issue, we need a method which can correctly identify both property and sentiment words.

In this work, we propose an approach that jointly analyzes the whole collection of product review snippets, induces a set of learned properties, and models the aggregate user sentiment towards these properties. We capture this idea using a Bayesian topic model where a set of properties and corresponding attribute tendencies are represented as hidden variables. The model takes product review snippets as input and explains how the observed text arises from the latent variables, thereby connecting text fragments with corresponding properties and attributes.

The advantages of this formulation are twofold. First, this encoding provides a common ground for comparing and aggregating review content in the presence of varied lexical realizations. For instance, this representation allows us to directly compare how many reviewers liked a given property of a product. Second, our model yields an efficient mean-field variational inference procedure which can be parallelized and run on a large number of review snippets.

We evaluate our approach in the domain of snippets taken from restaurant reviews on Yelp. In this collection, each restaurant has on average 29.8 snippets representing a wide spectrum of opinions about a restaurant. The evaluation we present demonstrates that the model can accurately retrieve clusters of review fragments that describe the same property, yielding 20% error reduction over a standalone clustering baseline. We also show that the model can effectively identify binary snippet attributes with 9.2% error reduction over applicable baselines, demonstrating that learning to identify attributes in the context of other product reviews yields significant gains. Finally, we evaluate our model on its ability to identify product properties for which there is significant sentiment disagreement amongst user snippets. This tests our model's capacity to jointly identify properties and assess attributes.

2 Related Work

Our work on review aggregation has connections to three lines of work in text analysis.

First, our work relates to research on extraction of product properties with associated sentiment from review text (Hu and Liu, 2004; Liu et al., 2005a; Popescu et al., 2005). These methods identify relevant information in a document using a wide range of methods such as association mining (Hu and Liu, 2004), relaxation labeling (Popescu et al., 2005) and supervised learning (Kim and Hovy, 2006). While our method also extracts product properties and sentiment, our focus is on multi-review aggregation. This task introduces new challenges which were not addressed in prior research that focused on per-document analysis.

A second related line of research is multi-document review summarization. Some of these methods directly apply existing domain-independent summarization methods (Seki et al., 2006), while others propose new methods targeted for opinion text (Liu et al., 2005b; Carenini et al., 2006; Hu and Liu, 2006; Kim and Zhai, 2009). For instance, these summaries may present contrastive view points (Kim and Zhai, 2009) or relay average sentiment (Carenini et al., 2006). The focus of this line of work is on how to select suitable sentences, assuming that relevant review features (such as numerical scores) are given. Since our emphasis is on multi-review analysis, we believe that the information we extract can benefit existing summarization systems.

Finally, a number of approaches analyze review documents using probabilistic topic models (Lu and Zhai, 2008; Titov and McDonald, 2008; Mei et al., 2007). While some of these methods focus primar-

ily on modeling ratable aspects (Titov and McDonald, 2008), others explicitly capture the mixture of topics and sentiments (Mei et al., 2007). These approaches are capable of identifying latent topics in the collection in opinion text (e.g., weblogs) as well as associated sentiment. While our model captures similar high-level intuition, it analyzes fine-grained properties expressed at the snippet level, rather than document-level sentiment. Delivering analysis at such a fine granularity requires a new technique.

3 Problem Formulation

In this section, we discuss the core random variables and abstractions of our model. We describe the generative models over these elements in Section 4.

Product: A product represents a reviewable object. For the experiments in this paper, we use restaurants as products.

Snippets: A snippet is a user-generated short sequence of tokens describing a product. Input snippets are deterministically taken from the output of the Sauper et al. (2010) system.

Property: A property corresponds to some fine-grained aspect of a product. For instance, the snippet "the pad thai was great" describes the *pad thai* property. We assume that each snippet has a single property associated with it. We assume a fixed number of possible properties K for each product.

For the corpus of restaurant reviews, we assume that the set of properties are specific to a given product, in order to capture fine-grained, relevant properties for each restaurant. For example, reviews from a sandwich shop may contrast the club sandwich with the turkey wrap, while for a more general restaurant, the snippets refer to sandwiches in general. For other domains where the properties are more consistent, it is straightforward to alter our model so that properties are shared across products.

Attribute: An attribute is a description of a property. There are multiple attribute *types*, which may correspond to semantic differences. We assume a fixed, pre-specified number of attributes N. For example, in the case of product reviews, we select $N = 2$ attributes corresponding to positive and negative sentiment. In the case of information extraction, it may be beneficial to use numeric and alphabetic types.

One of the goals of this work in the review domain is to improve sentiment prediction by exploiting correlations within a single property cluster. For example, if there are already many snippets with the attribute representing positive sentiment in a given property cluster, additional snippets are biased towards positive sentiment as well; however, data can always override this bias.

Snippets themselves are always observed; the goal of this work is to induce the latent property and attribute underlying each snippet.

4 Model

Our model generates the words of all snippets for each product in a collection of products. We use $s^{i,j,w}$ to represent the wth word of the jth snippet of the ith product. We use $\mathbf{s}$ to denote the collection of all snippet words. We also assume a fixed vocabulary of words V.

We present an overview of our generative model in Figure 1 and describe each component in turn:

Global Distributions: At the global level, we draw several unigram distributions: a global background distribution θ_B and attribute distributions θ_A^a for each attribute. The background distribution is meant to encode stop-words and domain white-noise, e.g., `food` in the restaurants domain. In this domain, the positive and negative attribute distributions encode words with positive and negative sentiments (e.g., *delicious* or *terrible*).

Each of these distributions are drawn from Dirichlet priors. The background distribution is drawn from a symmetric Dirichlet with concentration $\lambda_B = 0.2$. The positive and negative attribute distributions are initialized using seed words (V_{seed_a} in Figure 1). These seeds are incorporated into the attribute priors: a non-seed word gets ϵ hyperparameter and a seed word gets $\epsilon + \lambda_A$, where $\epsilon = 0.25$ and $\lambda_A = 1.0$.

Product Level: For the ith product, we draw property unigram distributions $\theta_P^{i,1}, \ldots, \theta_P^{i,K}$ for each of the possible K product properties. The property distribution represents product-specific content distributions over properties discussed in reviews of the product; for instance in the restaurant domains, properties may correspond to distinct menu items. Each $\theta_P^{i,k}$ is drawn from a symmetric Dirichlet prior

Global Level:

- Draw background distribution $\theta_B \sim \text{DIRICHLET}(\lambda_B V)$

- For each attribute type a,

  - Draw attribute distribution $\theta_A^a \sim \text{DIRICHLET}(\epsilon V + \lambda_A V_{seed_a})$

Product Level:

- For each product i,

  - Draw property distributions $\theta_P^k \sim \text{DIRICHLET}(\lambda_P V)$ for $k = 1, \ldots, K$

  - Draw property attribute binomial $\phi^{i,k} \sim \text{BETA}(\alpha_A, \beta_A)$ for $k = 1, \ldots, K$

  - Draw property multinomial $\psi^i \sim \text{DIRICHLET}(\lambda_M K)$

Snippet Level:

- For each snippet j in ith product,

  - Draw snippet property $Z_P^{i,j} \sim \psi^i$

  - Draw snippet attribute $Z_A^{i,j} \sim \phi^{Z_P^{ij}}$

  - Draw sequence of word topic indicators $Z_W^{i,j,w} \sim \Lambda | Z_W^{i,j,w-1}$

  - Draw snippet word given property $Z_P^{i,j}$ and attribute $Z_A^{i,j}$

$$s_{i,j,w} \sim \begin{cases} \theta_P^{i,Z_P^{i,j}}, & \text{when } Z_W^{i,j,w} = P \\ \theta_A^{Z_A^{i,j}}, & \text{when } Z_W^{i,j,w} = A \\ \theta_B, & \text{when } Z_W^{i,j,w} = B \end{cases}$$

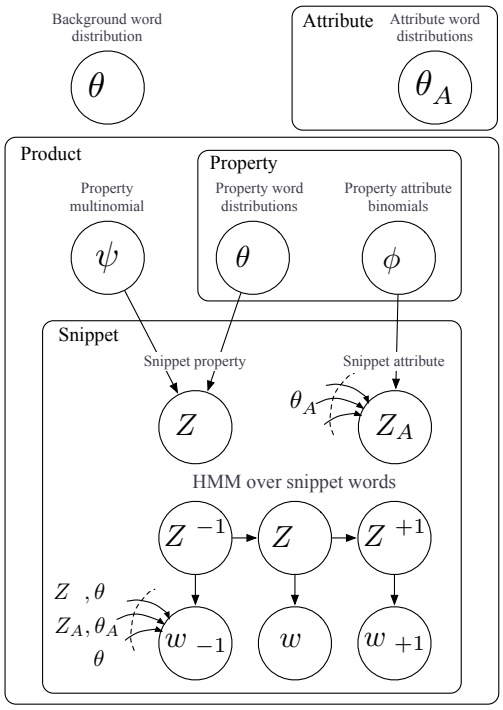

Figure 1: A high-level verbal and graphical description for our model in Section 4. We use $\text{DIRICHLET}(\lambda V)$ to denote a finite Dirichlet prior where the hyper-parameter counts are a scalar times the unit vector of vocabulary items. For the global attribute distribution, the prior hyper-parameter counts are ϵ for all vocabulary items and λ_A for V_{seed_a}, the vector of vocabulary items in the set of seed words for attribute a.

with hyper-parameter $\lambda_P = 0.2$.

For each property $k = 1, \ldots, K$. $\phi_{i,k}$, we draw a binomial distribution $\phi_{i,k}$. This represents the distribution over positive and negative attributes for that property; it is drawn from a beta prior using hyper-parameters $\alpha_A = 2$ and $\beta_A = 2$. We also draw a multinomial ψ^i over K possible properties from a symmetric Dirichlet distribution with hyper-parameter $\lambda_M = 1,000$. This distribution is used to draw snippet properties.

Snippet Level: For the jth snippet of the ith product, a property random variable $Z_P^{i,j}$ is drawn according to the multinomial ψ^i. Conditioned on this choice, we draw an attribute $Z_A^{i,j}$ (positive or negative) from the property attribute distribution $\phi^{i,Z_P^{j,j}}$.

Once the property $Z_P^{i,j}$ and attribute $Z_A^{i,j}$ have been selected, the tokens of the snippet are generated using a simple HMM. The latent state underlying a token, $Z_W^{i,j,w}$, indicates whether the wth word comes from the property distribution, attribute distribution, or background distribution; we use P, A, or B to denote these respective values of $Z_W^{i,j,w}$.

The sequence $Z_W^{i,j,1}, \ldots, Z_W^{i,j,m}$ is generated using a first-order Markov model. The full transition parameter matrix Λ parametrizes these decisions. Conditioned on the underlying $Z_W^{i,j,w}$, a word, $s^{i,j,w}$ is drawn from $\theta_P^{i,j}$, $\theta_A^{i,Z_P^{i,j}}$, or θ_B for the values P,A, or B respectively.

5 Inference

The goal of inference is to predict the snippet property and attribute distributions over each snippet given all the observed snippets $P(Z_P^{i,j}, Z_A^{i,j} | \mathbf{s})$ for all products i and snippets j. Ideally, we would like to marginalize out nuisance random variables and distributions. Specifically, we approximate the full

model posterior using variational inference:[1]

$$P(\psi, \boldsymbol{\theta_P}, \theta_B, \boldsymbol{\theta_A}, \phi, |\mathbf{s}) \approx Q(\psi, \boldsymbol{\theta_P}, \theta_B, \boldsymbol{\theta_A}, \phi)$$

where ψ, $\boldsymbol{\theta_P}$, ϕ denote the collection of latent distributions in our model. Here, we assume a full mean-field factorization of the variational distribution; see Figure 2 for the decomposition. Each variational factor $q(\cdot)$ represents an approximation of that variable's posterior given observed random variables. The variational distribution $Q(\cdot)$ makes the (incorrect) assumption that the posteriors amongst factors are independent. The goal of variational inference is to set factors $q(\cdot)$ so that it minimizes the KL divergence to the true model posterior:

$$\min_{Q(\cdot)} KL(P(\psi, \boldsymbol{\theta_P}, \theta_B, \boldsymbol{\theta_A}, \phi, |\mathbf{s}) \| Q(\psi, \boldsymbol{\theta_P}, \theta_B, \boldsymbol{\theta_A}, \phi)$$

We optimize this objective using coordinate descent on the $q(\cdot)$ factors. Concretely, we update each factor by optimizing the above criterion with all other factors fixed to current values. For instance, the update for the factor $q(Z_W^{i,j,w})$ takes the form:

$$q(Z_W^{i,j,w}) \leftarrow \mathbb{E}_{Q/q(Z_W^{i,j,w})} \lg P(\psi, \boldsymbol{\theta_P}, \theta_B, \boldsymbol{\theta_A}, \phi, \mathbf{s})$$

The full factorization of $Q(\cdot)$ and updates for all random variable factors are given in Figure 2. Updates of parameter factors are omitted; however these are derived through simple counts of the Z_A, Z_P, and Z_W latent variables. For related discussion, see Blei et al. (2003).

6 Experiments

In this section, we describe in detail our data set and present three experiments and their results.

Data Set Our data set consists of snippets from Yelp reviews generated by the system described in Sauper et al. (2010). This system is trained to extract snippets containing short descriptions of user sentiment towards some aspect of a restaurant.[2] We select only the snippets labeled by that system as referencing *food*, and we ignore restaurants with fewer than 20 snippets. There are 13,879 snippets in total, taken from 328 restaurants in and around the Boston/Cambridge area. The average snippet length is 7.8 words, and there are an average of 42.1 snippets per restaurant, although there is high variance in number of snippets for each restaurant. Figure 3 shows some example snippets.

The [P noodles] and the [P meat] were actually [+ pretty good].
I [+ recommend] the [P chicken noodle pho].
The [P noodles] were [- soggy].
The [P chicken pho] was also [+ good].

The [P spring rolls] and [P coffee] were [+ good] though.
The [P spring roll wrappers] were a [- little dry tasting].
My [+ favorites] were the [P crispy spring rolls].
The [P Crispy Tuna Spring Rolls] are [+ fantastic]!

The [P lobster roll] my mother ordered was [- dry] and [- scant].
The [P portabella mushroom] is my [+ go-to] [P sandwich].
The [P bread] on the [P sandwich] was [- stale].
The slice of [P tomato] was [- rather measly].

The [P shumai] and [P California maki sushi] were [+ decent].
The [P spicy tuna roll] and [P eel roll] were [+ perfect].
The [P rolls] with [P spicy mayo] were [- not so great].
I [+ love] [P Thai rolls].

Figure 3: Example snippets from our data set, grouped according to property. Property words are labeled **P** and colored blue, NEGATIVE attribute words are labeled **-** and colored red, and POSITIVE attribute words are labeled **+** and colored green. The grouping and labeling are *not* given in the data set and must be learned by the model.

For sentiment attribute seed words, we use 42 and 33 words for the positive and negative distributions respectively. These are hand-selected based on the restaurant review domain; therefore, they include domain-specific words such as *delicious* and *gross*.

Tasks We perform three experiments to evaluate our model's effectiveness. First, a cluster prediction task is designed to test the quality of the learned property clusters. Second, an attribute analysis task will evaluate the sentiment analysis portion of the model. Third, we present a task designed to test whether the system can correctly identify properties which have conflicting attributes, which tests both clustering and sentiment analysis.

[1] See Liang and Klein (2007) for an overview of variational techniques.

[2] For exact training procedures, please reference that paper.

Mean-field Factorization

$$Q(\boldsymbol{\psi}, \boldsymbol{\theta_P}, \theta_B, \boldsymbol{\theta_A}, \boldsymbol{\phi}) = q(\theta_B) \left(\prod_{a=1}^{N} q(\theta_A^a) \right) \left(\prod_{i}^{n} \left(\prod_{k=1}^{K} q(\theta_P^{i,k}) q(\phi^{i,k}) \right) \left(\prod_{j} q(Z_A^{i,j}) q(Z_P^{i,j}) \prod_{w} q(Z_W^{i,j,w}) \right) \right)$$

Snippet Property Indicator

$$\lg q(Z_P^{i,j} = k) \propto \mathbb{E}_{q(\psi^i)} \lg \psi^i(p) + \sum_{w} q(Z_W^{i,j,w} = P) \mathbb{E}_{q(\theta_P^{i,k})} \lg \theta_P^{i,k}(s^{i,j,w}) + \sum_{a=1}^{N} q(Z_A^{i,j} = a) \mathbb{E}_{q(\phi^{i,k})} \lg \phi^{i,k}(a)$$

Snippet Attribute Indicator

$$\lg q(Z_A^{i,j} = a) = \sum_{k} q(Z_P^{i,j} = k) \mathbb{E}_{q(\phi^{i,k})} \lg \phi^{i,k}(a) + \sum_{w} q(Z_W^{i,j,w} = A) \mathbb{E}_{q(\theta_A^a)} \lg \theta_A^a(s^{i,j,w})$$

Word Topic Indicator

$$\lg q(Z_W^{i,j,w} = P) \propto \lg P(Z_W = P) + \sum_{k} q(Z_P^{i,j} = k) \mathbb{E}_{q(\theta_P^{i,k})} \lg \theta_P^{i,j}(s^{i,j,w})$$

$$\lg q(Z_W^{i,j,w} = A) \propto \lg P(Z_W = A) + \sum_{a \in \{+,-\}} q(Z_A^{i,j} = a) \mathbb{E}_{q(\theta_A^a)} \lg \theta_A^a(s^{i,j,w})$$

$$\lg q(Z_W^{i,j,w} = B) \propto \lg P(Z_W = B) + \mathbb{E}_{q(\theta_B)} \lg \theta_B(s^{i,j,w})$$

Figure 2: The mean-field variational algorithm used during learning and inference to obtain posterior predictions over snippet properties and attributes, as described in Section 5. Mean-field inference consists of updating each of the latent variable factors as well as a straightforward update of latent parameters in round robin fashion.

6.1 Cluster prediction

The goal of this task is to evaluate the quality of property clusters; specifically the $Z_P^{i,j}$ variable in Section 4. In an ideal clustering, the predicted clusters will be cohesive (i.e., all snippets predicted for a given property are related to each other) and comprehensive (i.e., all snippets which are related to a property are predicted for it). For example, a snippet will be assigned the property *pad thai* if and only if that snippet mentions some aspect of the pad thai.

Annotation For this task, we use a set of gold clusters over 3,250 snippets across 75 restaurants collected through Mechanical Turk. In each task, a worker was given a set of 25 snippets from a single restaurant and asked to cluster them into as many clusters as they desired, with the option of leaving any number unclustered. This yields a set of gold clusters and a set of unclustered snippets. For verification purposes, each task was provided to two different workers. The intersection of both workers' judgments was accepted as the gold standard, so the model is not evaluated on judgments which disagree. In total, there were 130 unique tasks, each of which were provided to two workers, for a total output of 210 generated clusters.

Baseline The baseline for this task is a clustering algorithm weighted by TF*IDF over the data set as implemented by the publicly available CLUTO package.[3] This baseline will put a strong connection between things which are lexically similar. Because our model only uses property words to tie together clusters, it may miss correlations between words which are not correctly identified as property words. The baseline is allowed 10 property clusters per restaurant.

We use the MUC cluster evaluation metric for this task (Vilain et al., 1995). This metric measures the number of cluster merges and splits required to recreate the gold clusters given the model's output.

[3] Available at http://glaros.dtc.umn.edu/gkhome/cluto/cluto/overview with agglomerative clustering, using the cosine similarity distance metric.

	Precision	Recall	F1
Baseline	80.2	61.1	69.3
Our model	72.2	79.1	75.5

Table 2: Results using the MUC metric on the cluster prediction task. Note that while the precision of the baseline is higher, the recall and overall F1 of our model outweighs that. While MUC has a deficiency in that putting everything into a single cluster will artificially inflate the score, parameters on our model are set so that the model uses the same number of clusters as the baseline system.

Therefore, it can concisely show how accurate our clusters are as a whole. While it would be possible to artificially inflate the score by putting everything into a single cluster, the parameters on our model and the likelihood objective are such that the model prefers to use all available clusters, the same number as the baseline system.

Results Results for our cluster prediction task are in Table 2. While our system does suffer on precision in comparison to the baseline system, the recall gains far outweigh this loss, for a total error reduction of 20% on the MUC measure.

The most common cause of poor cluster choices in the baseline system is its inability to distinguish property words from attribute words. For example, if many snippets in a given restaurant use the word *delicious*, there may end up being a cluster based on that alone. Because our system is capable of distinguishing which words are property words (i.e., words relevant to clustering), it can choose clusters which make more sense overall. We show an example of this in Table 3.

6.2 Attribute analysis

We also evaluate the system's predictions of snippet attribute using the predicted posterior over the attribute distribution for the snippet (i.e., $Z_A^{i,j}$). For this task, we consider the binary judgment to be simply the one with higher value in $q(Z_A^{i,j})$ (see Section 5). The goal of this task is to evaluate whether our model correctly distinguishes attribute words.

Annotation For this task, we use a set of 260 total snippets from the Yelp reviews for 30 restaurants, evenly split into a training and test sets of 130 snippets each. These snippets are manually labeled POSITIVE or NEGATIVE. Neutral snippets are ignored for the purpose of this experiment.

The martini selection looked delicious The s'mores martini sounded excellent
The martinis were good The martinis are very good
The mozzarella was very fresh The fish and various meets were very well made

The best carrot cake I've ever eaten Carrot cake was deliciously moist
The carrot cake was delicious.
It was rich, creamy and delicious. The pasta Bolognese was rich and robust.

Table 3: Example phrases from clusters in both the baseline and our model. For each pair of clusters, the dashed line indicates separation by the baseline model, while the solid line indicates separation by our model. In the first example, the baseline mistakenly clusters some snippets about *martinis* with those containing the word *very*. In the second example, the same occurs with the word *delicious*.

Baseline We use two baselines for this task, one based on a standard discriminative classifier and one based on the seed words from our model.

The DISCRIMINATIVE baseline for this task is a standard maximum entropy discriminative binary classifier over unigrams. Given enough snippets from enough unrelated properties, the classifier should be able to identify that words like *great* indicate positive sentiment and those like *bad* indicate negative sentiment, while words like *chicken* are neutral and have no effect.

The SEED baseline simply counts the number of words from the positive and negative seed lists used by the model, V_{seed_+} and V_{seed_-}. If there are more words from V_{seed_+}, the snippet is labeled positive, and if there are more words from V_{seed_-}, the snippet is labeled negative. If there is a tie or there are no seed words, we split the prediction. Because the seed word lists are specifically slanted toward restaurant reviews (i.e., they contain words such as *delicious*), this baseline should perform well.

Results For this experiment, we measure the overall classification accuracy of each system (see Table

	Accuracy
DISCRIMINATIVE baseline	75.9
SEED baseline	78.2
Our model	80.2

Table 4: Attribute prediction accuracy of the full system compared to the DISCRIMINATIVE and SEED baselines. The advantage of our system is its ability to distinguish property words from attribute words in order to restrict judgment to only the relevant terms.

The naan was **hot** and **fresh**
All the veggies were really **fresh** and **crisp**.
Perfect mix of **fresh flavors** and **comfort** food
The lo main smelled and tasted **rancid**
My grilled cheese sandwich was a little **gross**

Table 5: Examples of sentences correctly labeled by our system but incorrectly labeled by the DISCRIMINATIVE baseline; the key sentiment words are highlighted. Notice that these words are not the most common sentiment words; therefore, it is difficult for the classifier to make a correct generalization. Only two of these words are seed words for our model (*fresh* and *gross*).

4). Our system outperforms both supervised baselines.

As in the cluster prediction case, the main flaw with the DISCRIMINATIVE baseline system is its inability to recognize which words are relevant for the task at hand, in this case the attribute words. By learning to separate attribute words from the other words in the snippets, our full system is able to more accurately judge their sentiment. Examples of these cases are found in Table 5.

The obvious flaw in the SEED baseline is the inability to pre-specify every possible sentiment word; our model's performance indicates that it is learning something beyond just these basic words.

6.3 Conflict identification

Our final task requires both correct cluster prediction and correct sentiment judgments. In many domains, it is interesting to know not only whether a product is rated highly, but also whether there is conflicting sentiment or debate. In the case of restaurant reviews, it is relevant to know whether the dishes are consistently good or whether there is some variation in quality.

Judgment P	Judgment A		Attribute / Snippet
Yes	Yes	-	The salsa isn't great
		+	Chips and salsa are sublime
		-	The grits were good, but not great.
		+	Grits were the perfect consistency
		-	The tom yum kha was bland
		+	It's the best Thai soup I ever had
		-	The naan is a bit doughy and undercooked
		+	The naan was pretty tasty
		-	My reuben was a little dry.
		+	The reuben was a good reuben.
Yes	No	-	Belgian frites are crave-able
		+	The frites are very, very good.
No	Yes	-	The blackened chicken was meh
		+	Chicken enchiladas are yummy!
		-	The taste overall was mediocre
		+	The oysters are tremendous
No	No	-	The cream cheese wasn't bad
		+	Ice cream was just delicious

Table 6: Example property-attribute correctness for the conflict identification task, over both property and attribute. Property judgment (P) indicates whether the snippets are discussing the same item; attribute judgment (A) indicates whether there is a correct difference in attribute (sentiment), regardless of properties.

To evaluate this, we examine the output clusters which contain predictions of both positive and negative snippets. The goal is to identify whether these are true conflicts of sentiment or there was a failure in either property clustering or attribute classification.

For this task, the output clusters are manually annotated for correctness of both property and attribute judgments, as in Table 6. As there is no obvious baseline for this experiment, we treat it simply as an analysis of errors.

Results For this task, we examine the accuracy of conflict prediction, both with and without the correctly identified properties. The results by property-attribute correctness are shown in Table 7. From these numbers, we can see that 50% of the clusters are correct in both property (cohesiveness) and attribute (difference in sentiment) dimensions.

Overall, the properties are correctly identified (subject of NEG matches the subject of POS) 68% of the time and a correct difference in attribute is identified 67% of the time. Of the clusters which are correct in property, 74% show a correctly labeled

Judgment		
P	A	# Clusters
Yes	Yes	52
Yes	No	18
No	Yes	17
No	No	15

Table 7: Results of conflict analysis by correctness of property label (P) and attribute conflict (A). Examples of each type of correctness pair are show in in Table 6. 50% of the clusters are correct in both labels, and there are approximately the same number of errors toward both property and attribute.

difference in attribute.

7 Conclusion

We have presented a probabilistic topic model for identifying properties and attitudes of product review snippets. The model is relatively simple and admits an efficient variational mean-field inference procedure which is parallelized and can be run on a large number of snippets. We have demonstrated on multiple evaluation tasks that our model outperforms applicable baselines by a considerable margin.

Acknowledgments

The authors acknowledge the support of the NSF (CAREER grant IIS-0448168), NIH (grant 5-R01-LM009723-02), Nokia, and the DARPA Machine Reading Program (AFRL prime contract no. FA8750-09-C-0172). Thanks to Peter Szolovits and the MIT NLP group for their helpful comments. Any opinions, findings, conclusions, or recommendations expressed in this paper are those of the authors, and do not necessarily reflect the views of the funding organizations.

References

David M. Blei, Andrew Y. Ng, and Michael I. Jordan. 2003. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022.

Giuseppe Carenini, Raymond Ng, and Adam Pauls. 2006. Multi-document summarization of evaluative text. In *Proceedings of EACL*, pages 305–312.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of SIGKDD*, pages 168–177.

Minqing Hu and Bing Liu. 2006. Opinion extraction and summarization on the web. In *Proceedings of AAAI*.

Soo-Min Kim and Eduard Hovy. 2006. Automatic identification of pro and con reasons in online reviews. In *Proceedings of COLING/ACL*, pages 483–490.

Hyun Duk Kim and ChengXiang Zhai. 2009. Generating comparative summaries of contradictory opinions in text. In *Proceedings of CIKM*, pages 385–394.

P. Liang and D. Klein. 2007. Structured Bayesian nonparametric models with variational inference (tutorial). In *Proceedings of ACL*.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005a. Opinion observer: Analyzing and comparing opinions on the web. In *Proceedings of WWW*, pages 342–351.

Bing Liu, Minqing Hu, and Junsheng Cheng. 2005b. Opinion observer: analyzing and comparing opinions on the web. In *Proceedings of WWW*, pages 342–351.

Yue Lu and ChengXiang Zhai. 2008. Opinion integration through semi-supervised topic modeling. In *Proceedings of WWW*, pages 121–130.

Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: modeling facets and opinions in weblogs. In *Proceedings of WWW*, pages 171–180.

Ana-Maria Popescu, Bao Nguyen, and Oren Etzioni. 2005. OPINE: Extracting product features and opinions from reviews. In *Proceedings of HLT/EMNLP*, pages 339–346.

Christina Sauper, Aria Haghighi, and Regina Barzilay. 2010. Incorporating content structure into text analysis applications. In *Proceedings of EMNLP*, pages 377–387.

Yohei Seki, Koji Eguchi, Noriko K, and Masaki Aono. 2006. Opinion-focused summarization and its analysis at DUC 2006. In *Proceedings of DUC*, pages 122–130.

Ivan Titov and Ryan McDonald. 2008. A joint model of text and aspect ratings for sentiment summarization. In *Proceedings of ACL*, pages 308–316.

Marc Vilain, John Burger, John Aberdeen, Dennis Connolly, and Lynette Hirschman. 1995. A model-theoretic coreference scoring scheme. In *Proceedings of MUC*, pages 45–52.