Aspect-Specific Ranking of Product Reviews Using Topic Modeling

Aaron Shen
Brown University Department of Computer Science
Providence, RI 02912
awshen@cs.brown.edu
Advisor: Professor Eugene Charniak

Abstract

We examine the problem of ranking different aspects of a product through examination of its customer reviews. For instance, a restaurant review may contain distinct and possibly differing opinions on the food, decor, service, and price. We present a ranking system that uses Latent Dirichlet Allocation (LDA) and a database of opinion-oriented words to predict the aspect-specific sentiment of individual reviews. We evaluate the ranker on a set of test reviews and compare our results to previous work in this area.

1 Introduction

As the Internet continues to affect the way we do business, customer-generated reviews are becoming an increasingly important part of e-commerce. Collecting and presenting such information in a useful way to consumers is a core part of the business of companies such as Amazon, Yelp and Tripadvisor. The online shopping community has come to place a great deal of reliance on customer reviews.

Reviews are typically written by consumers who have previously purchased a product or used a service and describe the user's experience with that product or service. Web sites which collect reviews often make them widely accessible so potential customers can read them and use the information for their own purchasing decisions. A standard review format is a block of text consisting of anywhere from a few sentences to a few paragraphs, and a numeric rating (often from 1 to 5 "stars") that is meant to summarize in a single number the reviewer's sentiment toward the product.

Many popular commercial websites only allow reviews to have a single numeric rating. This number is meant to correspond with how the customer would rate the product overall. However, such a ranking scheme is of limited use to the consumer. Consumers are interested not only in the overall quality of a product, but also its quality in certain specific aspect areas. For example, someone looking at reviews of a new restaurant will probably be interested in distinct aspects such as the quality of the food, the appeal of the decor, the quality of the service provided, and the perceived value with respect to price. These factors can be weighted differently for each person. Differing consumer preferences for certain features may contribute to the observation that ratings in one "overall" aspect are typically bimodal [7].

This suggests that one overall rating may be less meaningful than a set of ratings specific to each aspect of interest. On websites that lack aspect-specific rankings, practically the only way for the customer to obtain aspect-specific information is by reading the actual text of the reviews (see example in Figure 1). This can be time consuming, and the result may be potentially skewed since a product may have potentially hundreds of reviews and the customer most likely desires to read only a small subset of them, which may not representative of the whole corpus.

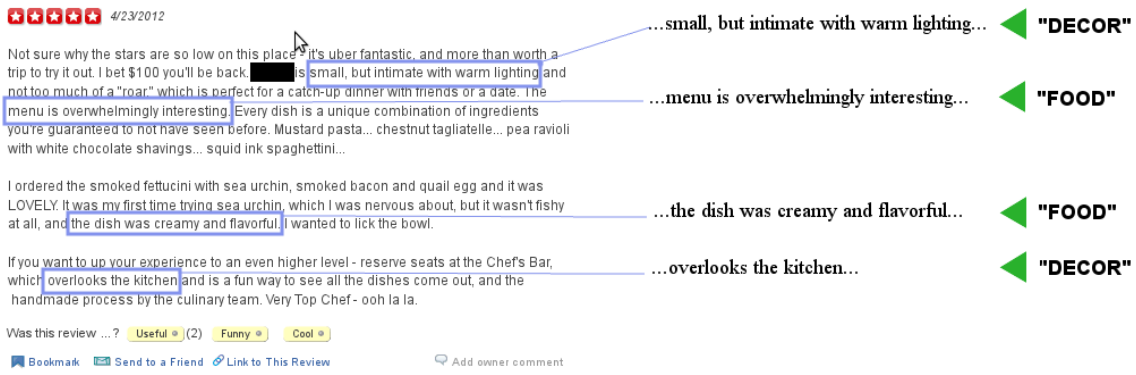

Figure 1: Example by-aspect parsing of a restaurant review. Review taken from Yelp.com.

In this paper, we introduce a ranker that, given s predefined aspects of interest, takes in the text of a review for a product and outputs a s-length vector of rankings in a set range (i.e. between 1 and 5 stars). Our model relies heavily on Latent Dirichet Allocation (LDA) [3], a popular topic model in computational linguistics. Research presented in [2] and [9] also provided some of the mechanisms behind the ranker.

The ranker first parses the reviews by using LDA to assign one or more aspects to each sentence of a review. This aspect information is then combined with aspect-neutral words in the text that appear in a lexicon of sentiment words (such a list was compiled in [6]) to define a list of r features and an r-length feature vector $\mathbf{x}$ for each review. For each aspect, a real-valued score is then calculated as $\mathbf{x} \cdot \mathbf{w}$, where $\mathbf{w}$ is an aspect-specific vector of weights learned from a set of training reviews. The scores are then mapped to rankings in discrete space based on thresholds learned during the training process.

The rest of this paper will describe and analyze the ranker in more detail. In Section 2 we go through some theory underlying the ranker and discuss related work. In Sections 3 and 4 we describe our experimental setup and assess the performance of the ranker on a real-world dataset. Performance is compared to a baseline and algorithms used in related studies. We conclude in Section 5 with an overall assessment and suggestions for further work.

2 Methodology

2.1 Aspect Definition

Hu and Liu [6] conducted early research in sentiment analysis of reviews by using a part-of-speech tagging approach to extract commonly occurring nouns as aspects and opinion-oriented words provide sentiment information. A later implementation of this model [8] showed that this approach yields aspects that are meaningful when evaluated by a test group of consumers. In these studies, using commonly occurring nouns as features sometimes yielded aspects that are very specific to each product (for example, in one example in [8], parsing reviews for video games produced the aspects ”flight simulator” and ”strategy game”).

Our approach is slightly different. We assume that for each class of products, there is a set of very general categorical aspects that are shared by all products in that class. The idea behind this is that since these aspects are common to all products, it will be easier for consumers to cross-compare between items. Using the aforementioned restaurant example, we could define the aspects, “food”,“decor”,“service”, and “price.” For other categories of products, aspects could be determined either through a survey of consumers, or by first using the approach in the two studies above then hand-picking aspects that are shared by all products in that genre.

The goal of the algorithm is to examine a the text of a review and determine a ranking for each predefined aspect that accurately reflects the sentiment in the text.

2.2 LDA

Because different sentences in the review may express opinions on different aspects, the first step in the algorithm is to use LDA to determine how much information each sentence in the review contains about the different aspects.

We apply a couple of modifications to the original LDA model as proposed in [3]. First, LDA has been shown to work poorly when the documents are individual sentences so we use the approach of sliding windows suggested by [9]. A sliding window is a group of consecutive sentences that share the same topic distribution and takes the place of the "document" concept in classic LDA. This helps to alleviate the (weak) assumption that sentence boundaries also represent topic transition boundaries.

This leads to the following generative model. First, for each of the topics in the model:

- Choose a distribution over words $\phi_{topic} \sim Dir(\alpha)$.

For each sentence s in a review doc:

- Choose a distribution over possible sliding windows $\Psi_{doc,s} \sim Dir(\beta)$

For each sliding window in the review:

- Choose a distribution over topics $\theta_{doc,sw} \sim Dir(\gamma)$

For each word i in sentence s of review doc:

- Choose window $v_{doc,i} \sim Cat(\Psi_{doc,s})$
- Choose topic $z_{doc,i} \sim Cat(\theta_{doc,v})$
- Choose word $w_{doc,i} \sim Cat(\phi_z)$

In order to sample from this distribution, we would like to obtain the marginalized distribution

$$P(\mathbf{w}, \mathbf{z}, \mathbf{v}) = P(\mathbf{w} \mid \mathbf{z})P(\mathbf{z} \mid \mathbf{v})P(\mathbf{v}) \tag{1}$$

[3] showed that a marginal distribution of this form cannot be computed directly, so we use Gibbs sampling as originally suggested by Griffiths and Steyvers [5]. The conditional distribution used to perform the Gibbs sampling for LDA without seeding is:

$$P(z_i = j, v_i = k \mid \mathbf{w}_{-i}, \mathbf{z}_{-i}, \mathbf{v}) \propto \left(\frac{n_{j \to w_i,-i} + \alpha}{n_{j,-i} + W\alpha}\right)\left(\frac{n_{k \to j,-i} + \gamma}{n_{k,-i} + T\gamma}\right)\left(\frac{n_{doc,s \to k,-i} + \beta}{n_{doc,s,-i} + V\beta}\right) \tag{2}$$

Here $n_{j \to w_i,-i}$, refers to the number of times topic j generates word w_i, $n_{j,-i}$ refers to the total count of words assigned to topic j, $n_{k \to j,-i}$, refers to the number of times window k generates topic j, $n_{k,-i}$ refers to the total count of words in window k, $n_{doc,s \to k-i}$, refers to the number of words in sentence s of review doc that are assigned to window k, and $n_{doc,s,-i}$ refers to the total words in sentence s. The $-i$ notation indicates that these quantities are to be calculated minus the topic/window assignment of the current word under consideration. Since the release of the Griffiths and Steyvers paper, the Gibbs sampler has become a popular method for doing LDA and the derivation of the basic formula is well-documented in the literature. Thus for brevity we do not include the derivation in the body of this paper. However, it is included as an appendix for reference purposes.

We also experimented with using predefined seed words in the LDA clustering process. This method is also referred to as topic-in-set knowledge and is well-summarized in [1]. It is reasonable to think that some words are more likely to be affiliated with certain aspects than others. Continuing with our restaurant example, we might theorize that the word "delicious" is more likely to be affiliated with the "food" aspect than the "decor," "service," or "value" aspects. Then if we choose topic j to be our "food" topic, we implement a rule that assigns the word "delicious" to topic j with high

Seed Words	Aspect
food menu delicious tasted	food
ambience decor clean setting	decor
service waiter waitress helpful	service
money paid cost expensive	value

Table 1: Seed words for restaurant example

probability. See Table 1 for an example. We theorized seeding might allow the topics to better match the aspects of interest, leading to a lower ranking loss. Seeding also limits the number of topics overall, which speeds up Gibbs sampling and also reduces the complexity of the feature set, leading to faster training (see Section 4).

To use seed words, we apply the mechanism in [1] of adding an adjustment term $\eta \mathbb{1}\{w_i \in C^j\} + 1 - \eta$ so that the distribution becomes

$$P(z_i = j, v_i = k \mid \mathbf{w}_{-i}, \mathbf{z}_{-i}, \mathbf{v}) \propto$$
$$\left(\frac{n_{j \to w_i, -i} + \alpha}{n_{j,-i} + W\alpha}\right)\left(\frac{n_{k \to j, -i} + \gamma}{n_{k,-i} + T\gamma}\right)\left(\frac{n_{doc,s \to k, -i} + \beta}{n_{doc,s,-i} + V\beta}\right)(\eta \mathbb{1}\{w_i \in C^v\} + 1 - \eta) \quad (3)$$

The condition $w_i \in C^j$ evaluates to true if and only if w_i is in our seed words for topic j. We vary η between 0 and 1 to specify the strength of the constraint. $\eta = 0$ is equivalent to unconstrained sampling while $\eta = 1$ is equivalent to a hard constraint.

We use this model to gain inference into sentences being about certain aspects. We run the Gibbs sampler for enough iterations to allow it to converge, then sample each sentence a fixed number of times and keep track of the topic assignments of each word in the sentence. We assume a sentence is x percent about a certain topic, if x percent of the samples from that sentence are assigned to that topic. Across all topics these percentages sum to one for each sentence. The percentages are stored for each individual sentence.

2.3 Feature Representation

Modeling reviews as vectors of lexical features is a common approach in sentiment classification. We create our set of features by bucketing the percentages from the previous step and concatenating the buckets with opinion words from the 6,789-word lexicon from provided by [6] (Table 2). These words, all unigrams, are various English parts of speech which are opinion-oriented but neutral with regard to subject. Therefore, they can be viewed as providing sentiment information but not topic information. The original lexicon is divided into positive sentiment and negative sentiment categories, however for our experiment we simply used the whole lexicon and did not make any distinction between the two categories.

Positive	Negative
a+	abnormal
abundant	abominable
accurate	abort
...	...
wow	wrong
yay	yawn
zenith	zealous

Table 2: Example words from opinion lexicon in [6]

We also considered the important problem of negation, that is, distinguishing between "great" in the context of "this was great" versus "this was not great." We addressed this by checking for the presence of negation words (such as not, wasn't, didn't) in the sentence and concatenating a binary flag (negated, not negated) as part of the feature. Opinion words that came 5 words or less after a negation word were considered to be negated. Other similar studies have negated all the words after a negation word, however more recent work has suggested 5 as a good "neighborhood" to use. [6]

Then, an example feature for the aspect "decor" would be:

review contains word **"nice"** and **not negated**
in sentence about **"decor"** with percentage **0.1-0.5**

Features are turned on or off in a review depending on whether the underlying combination of words and context is present or absent in the review.

Assuming r is the number of features in the model, s is the number of aspects we wish to rank, and t is the number of possible ratings (stars) for each aspect, we represent each review by a r-length 0-1 vector $\mathbf{x}$. A training set of reviews where the aspect-specific rankings are known is used to learn the r-length weight vectors $\mathbf{w}_1 \ldots \mathbf{w}_s$ and a s sets of $t-1$ thresholds (one for each aspect).

In unseeded LDA, we store the weight vectors as an $r \times s$ matrix $\mathbf{W}$. $\mathbf{xW}$ gives us a vector of aspect-specific scores and the thresholds are used to convert the real-valued scores to a ranking on a multi-point scale (i.e. 1 to 5 stars). A score translates to a rank of i if it is less than or equal to the ith threshold value for that aspect. The learned matrix $\mathbf{W}$ and thresholds can then be applied to novel review data for testing and classification.

In seeded LDA, rather than having one feature vector with all the topics, we store s separate feature vectors $\mathbf{x}_1 \ldots \mathbf{x}_s$, each containing information about a specific topic, Then $\mathbf{x}_1 \cdot \mathbf{w}_1 \ldots \mathbf{x}_s \cdot \mathbf{w}_s$ gives us the scores for each of the aspects and we proceed normally.

For the learning step we use the Good Grief algorithm, a perceptron-style algorithm presented in [2], which in turn relies on work done by [4].

2.4 Related Work

The procedure to go from LDA probabilities to rankings was suggested by Titov [9], who also used LDA in conjunction with the Good Grief algorithm in [2] to determine review rankings.

Our methodology differs from theirs in several aspects. Whereas [9] and [2] used commonly occurring unigrams, bigrams, and trigrams as features, we restrict our features to only those adjectives appearing in the lexicon from Hu and Liu's experiment described in section 2.1 [6]. The motivation for this is that some commonly occurring text in reviews may by itself provide topic-specific information (For example the phrase "tasted terrible" is highly likely to be associated with the "food" aspect in a restaurant review), and in these cases the additional gains to be had from topic modeling may be small. Our experimentation with pre-seeding of the LDA is another novel feature.

3 Implementation

To test our ranker we obtained a dataset consisting of archived hotel reviews from the website Tripadvisor.com [10] and ranked a selection of reviews for the aspects of "value," "room," "location," and "service." We selected this dataset because Tripadvisor is one of the few websites that allows users to rate specific categories, so the dataset contains actual rankings from users for these aspects. This allows us to compare our predicted rankings to ground truth. Tripadvisor reviews were also used in [9] which provides a further basis for comparison.

The seed words we chose for each aspect are in Table 3. We chose these by examining a list of most frequent words in the corpus, and subjectively choosing those that might be affiliated with a certain topic but not with others. Future experiments might use an objective criterion such as a TF-IDF calculation or the LARA model. [10]

price money value rate	value
room bathroom bed beds	room
location located area nearby	location
staff service friendly helpful	service

Table 3: Seed words used in 5 topic experiment

From the full dataset we selected 30,000 reviews randomly so that hotels of various qualities and locations would be represented. From this collection we randomly selected ten orderings of 22,000 reviews, using 20,000 reviews for training and 2,000 reviews for testing (with a separately selected parcel of reviews used for development).

We removed some common stopwords from each parcel of reviews and then then ran our Gibbs sampler separately on each parcel for 250 iterations, using five topics. We tested both seeded and unseeded LDA implementations. For the seeded LDA, we associated four topics with the seed words in Table 3 for plus an unseeded fifth topic to add flexibility to the model. The number and choice of seed words was done during development on a dataset separate from the training set. For the unseeded LDA, we used 11 topics using the the distribution of Equation Training and test reviews were sampled together during Gibbs sampling so the topics learned would be consistent. The resulting topic percentages for each sentence were turned into features as described in the previous section. We used three probability buckets corresponding to percentages of below 10%, 10-50%, and 50+%.

For each review, we then predicted rankings for the categories of value, room, location, service on a scale between 1 and 5 stars, and compare these with the actual rankings given by the user.

4 Results and Analysis

4.1 Results

Table 4 shows our results, which are given in ranking loss [4], which is calculated as

$$\sum_{i=0}^{N} \frac{|\text{actual\_rating}(i) - \text{predicted\_rating}(i)|}{N}$$

where N is the total number of rankings in the test set. We compare the ranking loss from our classifier to a baseline where the ranks for every review are predicted to be 5 stars. This ranking is the most common ranking in our dataset and has been used as the typical baseline for other studies in this area. For reference we also ran the dataset through our own implementation of the Good Grief classifier in [2] and include the results here. The numbers shown are averages across all ten test sets, and the "overall" aspect is simply the average of the four aspects. Lower ranking loss represents greater accuracy.

	Value	Room	Location	Staff	Overall
Baseline	1.1443	1.1392	0.7905	1.0113	1.0213
LDA + Opinion Words (seeded)	0.8056	0.8657	0.8606	0.8612	0.8483
LDA + Opinion Words (unseeded)	0.7775	0.8295	0.8088	0.7897	0.8013
PRank	0.7422	0.7419	0.7621	0.7514	0.7494

Table 4: Ranking loss for baseline and tested models.

Each of the tested models performed better overall than the baseline, and in all the individual aspects except for location. (Baseline location ranking loss lower than that of other aspects is consistent with the result in [9]. This leads us to conclude high location scores outweigh low location scores by a margin greater than for any other aspect.) Our results show clear gains relative to the baseline from using topic modeling, even when the feature words provide no additional aspect-specific information.

Performance using our LDA-opinion word models is comparable to that using PRank, which was about 0.05 stars better than the seeded LDA model. Performance using unseeded LDA was about 0.05 stars worse than the model using seeded LDA. We examined the topics that were created by seeded and unseeded LDA to see why this might be the case (Tables 5 and 6).

By a qualitative analysis, we see that the topics for seeded LDA did not necessarily correspond better to our aspects than the unseeded LDA. On the contrary, the unseeded LDA seemed to "learn" topics corresponding to the aspects on its own, without need for seeding. Unseeded LDA also exhibited cohesive themes for several other topics. Some of these auxiliary topics corresponded to

Representative words	Theme
town, center, road, stop, far, blocks, centre, 15, convenient, square, central, metro, train, short, it's, main, 5, shopping, bus, minute, distance, near, easy, 10, just, airport, area, located, restaurants, minutes, walking, station, street, city, close, right, location, away, hotel	location
poor, helped, attentive, gave, courteous, professional, warm, spoke, people, wonderful, customer, speak, rude, polite, welcome, kind, nice, needs, feel, special, efficient, happy, make, especially, really, guests, help, concierge, way, pleasant, reception, english, extremely, excellent, desk, hotel, helpful, friendly, service	service
kitchen, fridge, new, screen, pillows, quiet, flat, tub, decorated, single, amenities, 2, decor, bathrooms, king, bedroom, bath, quite, space, huge, towels, standard, double, suite, spacious, size, big, modern, tv, beds, shower, large, nice, comfortable, bed, bathroom, small, clean, rooms	room (amenities)
heard, closed, inside, conditioning, quiet, loud, smell, walls, doors, smoking, wall, looked, dirty, non, quite, elevator, bit, light, didn't, hear, windows, work, outside, window, noisy, street, problem, sleep, view, old, building, air, open, rooms, did, noise, night, door, floor	room (cleanliness/noise)
reasonable, book, park, prices, better, valet, need, cheap, hour, hotels, plus, wireless, deal, business, lot, 1, high, euros, cost, used, charge, 2, got, expensive, available, paid, extra, car, lobby, worth, pay, rate, access, price, parking, use, day, internet, night	value
places, expensive, cafe, try, continental, eggs, order, local, lunch, selection, served, plenty, lounge, choice, complimentary, ate, restaurants, quality, drink, fruit, wine, day, cold, evening, tea, included, fresh, morning, drinks, hot, eat, dinner, water, buffet, coffee, bar, restaurant, food, good	food

Table 5: Representative words for sample unsupervised topics.

Representative words	Aspect
family, week, say, 5, rate, service, excellent, 2, days, money, wonderful, new, staying, star, loved, reviews, business, better, definitely, value, overall, 4, 3, experience, best, booked, trip, just, night, time, recommend, good, nights, price, hotels, place, great, stayed, stay	value
looked, standard, quiet, open, window, building, night, space, sleep, huge, towels, double, 2, air, old, suite, door, little, spacious, quite, size, bit, modern, big, tv, noise, like, beds, shower, large, view, floor, comfortable, bathroom, bed, nice, small, clean, rooms	room
want, park, metro, convenient, train, far, quiet, nice, short, center, central, main, shopping, bus, minute, distance, near, 5, excellent, easy, 10, located, walking, airport, station, minutes, restaurants, close, away, right, area, good, "its", street, city, just, walk, great, location	location
hours, late, concierge, going, ask, checked, help, came, called, morning, early, english, took, don't, know, problem, parking, left, sure, said, way, asked, reception, told, arrived, didn't, went, make, service, got, night, people, check, time, day, did, desk, helpful, friendly	service

Table 6: Representative words for seeded topics.

our core aspects; for example, several adjectives in the "food" category below had high weights for the "value" aspect. These extra topics are most likely the reason unseeded LDA did better.

We examined the distribution of features in the reviews between the PRank model and our ranker, to examine why PRank outperformed our model. Sparsity of features appears to have been a contributing factor. Table 7 shows that PRank performed similarly to the LDA model for reviews which had less than 10 features turned on in its respective model. Because PRank uses commonly-used words as features, it had less reviews fall into this category: 1,840 reviews versus 3,078 reviews for

the LDA model using opinion words. This suggests we could improve the usefulness of this type of ranker by simply refusing to output aspect scores for a review unless a certain threshold of features is met or exceeded.

	Value	Room	Location	Staff	Overall
LDA + Opinion Words (unseeded)	0.8898	0.9594	0.8148	0.9012	0.8913
PRank	0.8918	0.9288	0.7908	0.8668	0.8696

Table 7: Ranking loss for baseline and LDA models for reviews < 10 features.

We then examined general ways our ranker might be improved relative to the baseline. Figure 2 shows the distribution of rating error for each aspect, for the unseeded LDA and PRank models. The rating error is calculated as (predicted rank $-$ true rank) so a rating error above zero indicates overprediction (i.e. predicting a ranking higher than the true ranking), while a rating error below zero indicates underprediction.

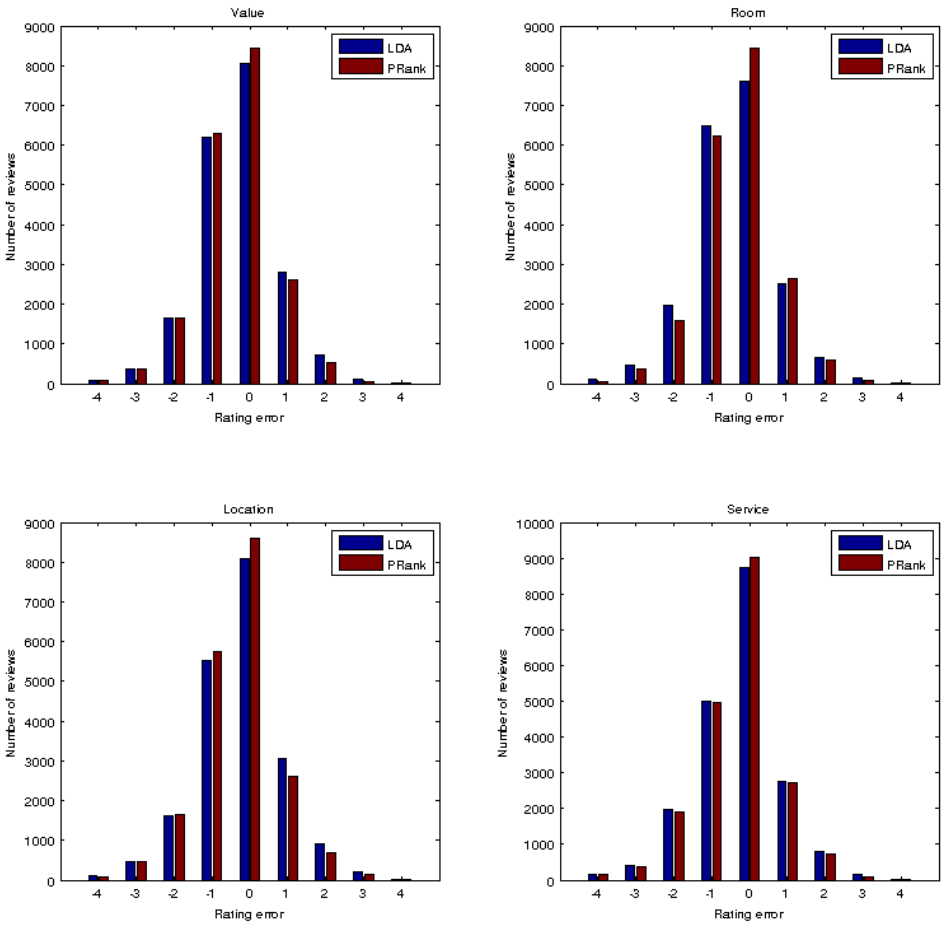

Figure 2: Distribution of rating error by aspect.

We note that in both models, and for all aspects, the distribution is roughly bell-shaped, with a substantial bias towards underprediction. Given that the most common rating consists of all 5s, we would expect the the natural distribution of ratings to be higher, with underprediction occurring as a result. This suggests that the ranker might be improved if it factored in some prior distribution on the

rankings, regardless of the text data, when making its prediction. This prior distribution could given in the aggregate for the entire corpus or even for the reviews specific to each individual product. Keeping a distribution on each individual product would allow for the ranker to better differentiate between products of varying quality. We note that our baseline has access to such prior knowledge, which explains much of its relatively good performance compared to our models.

Another source of error in both PRank and our ranker is the fact that they do not consider different people are likely to use the same opinion words corresponding to different ratings. For example, one reviewer might use the word "good" and give a hotel all 5's, while another might use "good" in a context of all 3's. A possible direction for future research might be to consider actual authorship of reviews in the ranker, or assign some latent variable to the reviewer's "standards" based on the word choices they use, and their ranking.

4.2 Speed Differences Between Models

We note that the seeded LDA, while performing at a lower level than unseeded LDA, achieved several times speedup from that model. Since the unseeded version only performs about 0.05 stars better, in some cases where speed is paramount, it may be better to use the seeded model.

The first speedup occurs during Gibbs sampling, which has a time complexity proportional to the number of topics used. As Gibbs sampling is usually run for hundreds of iterations, this alone can provide a substantial speedup.

Speedup also occurs during Good Grief training which can be explained by the length of the feature vector for each model. The number of features in the seeded model is $2\left|L\right|b$, where $\left|L\right|$ is the number of words in the lexicon and b is the number of buckets probabilities are assigned to. We multiply by 2 to account for negated versus non-negated features.

For comparison, the number of features in the unseeded model, which is $2\left|L\right|b * T$, where T is the number of topics used in the LDA. This means the seeded model improves speed and complexity by a factor of T, at the cost of some ranking loss.

The number of features in the seeded model (40,752 for a 6,792-word lexicon using 3 probability buckets) compares similarly to the number of features used in the base PRank algorithm (about 30,000 for a dataset of about 3,000 reviews in [2]) and compares favorably to the number of features used in [9], which used $\left|L'\right|Tb$ features, where T is the number of topics in used in the LDA and L' is the number of base bigrams, trigrams, and unigrams used for classification. PRank, however, does not have the overhead of a Gibbs sampling run.

5 Conclusion

We have presented a method of providing aspect-specific rankings for text reviews. We have discussed previous research in this field on which our ranker is based and tested some modifications to existing models, and also made some suggestions for future research.

Our results show that topic modeling can provide increased accuracy for aspect-specific rankings even when aspect-neutral words are used are features. Using seed words in the topic model can decrease the size of the feature set and lead to faster training times for the classification algorithm, with the tradeoff of slightly lower accuracy.

6 Acknowledgments

I would like to thank my advisor, Dr. Eugene Charniak, for all his help on this project. His availability, advice, and insight were essential to its completion.

References

[1] D. Andrezejewski and X. Zhu. Latent Dirichlet allocation with topic-in-set knowledge. 2009. *Proc. of the NAACL HLT 2009 Workshop on Semi-Supervised Learning for Natural Language Processing*

[2] B. Snyder and R. Barzilay. Multiple aspect ranking using the Good Grief Algorithm. 2007. *Proc. of the Joint Conference of the North American Chapter of the Association for Computation Linguistics and Human Language Technologies.*

[3] D. Blei, A. Ng, and M. Jordan. Latent Dirichlet allocation. 2003. *Journal of Machine Learning Research.*

[4] K. Crammer and Y. Singer. Pranking with ranking. 2001. *Advances in Neural Information Processing Systems 14.*

[5] T.L. Griffiths and M. Steyvers. Finding scientific topics. 2004. *Proc. of the Natural Academy of Sciences.*

[6] M. Hu and B. Liu. Mining and summarizing customer reviews. 2004. *Proc. of the 10th KDD*

[7] N. Hu, Pavlou, P., Zhang, J. Can Online Reviews Reveal a Product's True Quality? Empirical Findings and Analytical Modeling of Online Word-of-Mouth Communication. 2006. *Proc. 7th ACM Conf. on Electronic Commerce*

[8] Scaffidi, Bierhoff, et al. Red Opal: Product-feature scoring from reviews. 2007. *EC'07.*

[9] I. Titov and R. McDonald. Modeling online reviews with multi-grain topic models, *WWW 2008.*

[10] H. Wang, Y. Lu, and C.X. Zhai. Latent aspect rating analysis without aspect keyword supervision 2001 *KDD'2011*

A Appendix 1

The marginalized distribution $P(\mathbf{w}, \mathbf{z}, \mathbf{v})$ is made up of three symmetric marginalized distributions.

$$P(\mathbf{w}, \mathbf{z}, \mathbf{v}) = P(\mathbf{w} \mid \mathbf{z})P(\mathbf{z} \mid \mathbf{v})P(\mathbf{v}) \tag{4}$$

Thus we can approach the marginalization one term at a time. Using the formula for the Dirichlet prior and integrating out β from $P(\mathbf{v}, \beta)$ gives

$$P(\mathbf{v}) = \left(\frac{\Gamma(V\beta)}{\Gamma(\beta)^V}\right)^{N_s} \prod_{s=1}^{N_s} \frac{\prod_{v=1}^{V} \Gamma(n_{doc,s\to v} + \beta)}{\Gamma(n_{doc,s} + V\beta)} \tag{5}$$

where V is the number of sliding windows available to sentence s, N_s is the number of sentences in the corpus, $n_{doc,s\to v}$ is the count of words in sentences s assigned to window v, and $n_{doc,s}$ is the total count of words in the sentence.

Similarly, the equations for the second and first terms are

$$P(\mathbf{z} \mid \mathbf{v}) = \left(\frac{\Gamma(T\gamma)}{\Gamma(\gamma)^T}\right)^{N_v} \prod_{v=1}^{N_v} \frac{\prod_{z=1}^{T} \Gamma(n_{v\to doc,z} + \gamma)}{\Gamma(n_{doc,v} + T\gamma)} \tag{6}$$

$$P(\mathbf{w} \mid \mathbf{z}) = \left(\frac{\Gamma(W\alpha)}{\Gamma(\alpha)^W}\right)^{T} \prod_{z=1}^{T} \frac{\prod_{w=1}^{W} \Gamma(n_{z\to w} + \alpha)}{\Gamma(n_z + W\alpha)} \tag{7}$$

In (3), N_v is the number of sliding windows in the corpus, $n_{doc,v\to z}$ is the count of times window v generates topic z, and $n_{doc,v}$ is the total count of words in window v. In (4), T is the number of topics in the model, $n_{z\to w}$ is the count of times topic z generates word w, and n_z is the total count of words assigned to topic z.

The conditional distribution $P(z_i, v_i \mid \mathbf{w}_{-i}, \mathbf{z}_{-i}, \mathbf{v})$ can then be derived through cancellation of terms (2)-(4).

$$P(z_i = j, v_i = k \mid \mathbf{w}_{-i}, \mathbf{z}_{-i}, \mathbf{v}) \propto \left(\frac{n_{j\to w_i,-i} + \alpha}{n_{j,-i} + W\alpha}\right)\left(\frac{n_{k\to j,-i} + \gamma}{n_{k,-i} + T\gamma}\right)\left(\frac{n_{doc,s\to k,-i} + \beta}{n_{doc,s,-i} + V\beta}\right) \tag{8}$$

In this equation $n_{j\to w_i,-i}$, $n_{j,-i}$, $n_{k\to j,-i}$, $n_{k,-i}$, $n_{doc,s\to k-i}$, and $n_{doc,s,-i}$ refer to their counterparts in equations (5)-(7), without regard to the topic and window assignments of the current word being considered.