Great Food, Lousy Service:
Topic Modeling for Sentiment Analysis in Sparse Reviews

Final Project
CS 224N / Ling 284 — Spring 2010

Robin Melnick
rmelnick@stanford.edu

Dan Preston
dpreston@stanford.edu

The Open Table challenge: Reconciling sparse text and disparate ratings!

1. INTRODUCTION

One star for "Service," but "They never disappoint!"? Does this guy have incredibly low expectations for being waited on or what! "Great food, lousy service, but hey, we love the place!" Or at least this is what the ratings seem to say. Still, even this reasonable prose reconciliation of the disparate topic scores is not at all what's explicitly in the text, just "They never disappoint!" In fact, the text doesn't speak to the topic ratings at all. Such, it seems, are among the multiple challenges of attempting Sentiment Analysis (SA) with data from OpenTable.com, a web-based restaurant reservation and rating service.

The point here is that while sentiment analysis is somewhat well-trod ground—at least to the extent that in the world of NLP, 10 years of research is what passes for "well-trod," where in many other fields, 10 years barely qualifies as "infancy"—the unique environment of short-format, multiple-rating restaurant reviews from OpenTable in fact presents several particular challenges:

1. The reviews are short: 248 characters on average.
2. There are four sub-topic ratings in addition to an overall score.
3. Ratings are scalar (one to five stars), as opposed to a binary thumbs-up/thumbs-down.
4. User reviewers are under no obligation to make their text and scores correlate in any principled or systematic way.
5. While reviewers give a 1-5 rating for each sub-topic, the text of many reviews actually says little—or sometimes nothing!—specific to these sub-topics.

It's these several challenges that motivated us to attempt more than 30 different classifier features—some major distinct architectural attempts, some minor incremental shifts—trialed empirically in nearly 50 different groupings, as we sought to deliver the best possible performance, optimized across several dimensions of scoring.

Given the apparent substantially loose relationship between text and ratings, it initially proved difficult to make truly dramatic increases in performance over baseline on the several metrics we evaluated—though perhaps "dramatic" here is a relative term. Ultimately, in fact, we're quite pleased that through several of our features we were able to eke out steady, measured progress on these metrics. The bulk of the paper will be devoted to discussion of these features and their generally data-driven motivations, with the final—and indeed perhaps "dramatic"—of these being Entropy-based features that reflect the rating shape or curve for a number of individual words.

a. Prior Work in Sentiment Analysis

Pang, Lee, and Vaithyanathan 2002 lays out the general techniques of SA as applied to movie reviews and compares different machine-learning engines for use in such efforts, including Naïve Bayes, maximum entropy classification (MEMM), and support vector machines (SVM). (Though each has advantages, the researchers settled on an SVM as the overall best performer for their SA.) Among other elements, they also introduce an innovative negation-handling process that we borrow for our work. In Pang and Lee 2004, the authors subdivide text in an attempt to look only at subjective portions. Though there is no sense of within-document classification for topic, theirs constitutes a first step beyond Bag of Words (BoW) to look only at certain portions of a text and is thus a precursor to the sub-topic modeling—*food, service, ambiance, noise, overall score*—that we take on in the present work. Snyder and Barzilay 2007 look at another set of (longer) restaurant reviews—an initial attempt at addressing sub-topic ratings.

Further important SA work exists in an unpublished study by Chris Potts (personal communication), in which he explores rating correlations for individual sentiment words in a manner not dissimilar to a portion of the analysis we undertake.[1]

b. Collaborative Project Background

While the NLP innovations of the present study are the work of the authors, this project was initially spun out from an effort initiated by Andrew Maas. He gained access to the data and set up a simple unigram (Bag of Words/BoW) implementation within a basic SVM framework. From there, he was eager to collaborate with others to see what language-infused features could do. We are one of three CS224N groups working within this framework.

2. DATA

The first step in addressing the system is to try to well understand, visualize, and analyze the distribution of the available ratings data. The complete data consists of 456,983 reviews of 11,067 different restaurants serviced by OpenTable.com. The data is made available in Google Protocol Buffer format.

a. Structure

Each review includes text and ratings. The text is limited to 750 chars. As compared to Twitter (140 character limit) then, these reviews fall into an intermediate range between Tweets and full-length prose reviews. There are no restrictions or enforcements made on content entered, so in practice reviews range dramatically in length even within this 750-character range—with a minimum length in this data set of 1 (a single character) and a maximum, of course, of 750. Note the high standard deviation relative to the mean:

	# of chars
Min	1
Max	750
Avg	248
Std Dev	190

[1] Though the exploration of individual word-to-ratings correlation curves/shapes is similar, the Entropy-based classifier feature we introduce herein to exploit this information is original to the present work.

a. Sparsity

Beyond the text, each review has five numerical ratings: ***Overall***, ***Food***, ***Ambiance***, ***Service***, and ***Noise***. (Each is on a scale of 1 to 5 stars, 5 being the best, with the exception of ***Noise***, which is rated on a 3-point scale.) Significantly, every reviewer provides scores on each of these but without necessarily saying anything in the text about each of them. Or any of them! We found many examples where the text does not mention a particular category at all—the snippet presented at the top of the paper being just one of the more spectacular examples—and it is most common that a given review will have comparatively more commentary on one or another of the categories.

Just a few of these examples of sparsity:

(1) *The brunch was excellent. We all had great time.*
(We can take *brunch* here as a marker for *food* topic, but *service, ambiance, noise*?)

(2) *Excellent food and our waiter was outstanding.*
(Here we get *food* and *service*, but how about *ambiance* and *noise*?)

(3) *An unexpected combination of Left-Bank Paris and Lower Manhattan in Omaha. Divine. Inspirational and a great value.*
(This is our favorite! Sounds lovely, but not very helpful on, say, *noise*!)

b. Ratings Distribution

Prior published work, meetings with course staff, and personal communication with Chris Potts all led us to expect the ratings data to be somewhat skewed, though the nature of that skewing remained to be seen. It was suggested, for example, that we might see a strong bias towards positive reviews, but we might also see a bimodal distribution, with peaks at both high and low ends. As anticipated, there is in fact a skewing, and here it turns out to be towards a single peak at the high end.

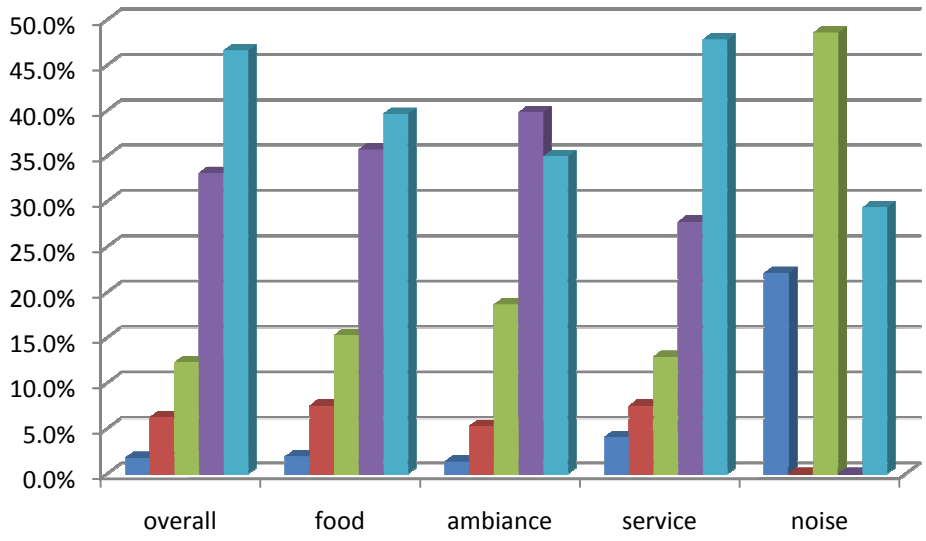


Figure 1: overall and sub-topic rating distributions

As the first set of columns in Figure 1 shows, the skewing towards the high end is extreme, with more than 75% of all reviews given a 4 or 5 overall. The sub-topic ratings also all skew towards the high end—if slightly less dramatically than ***Overall***—with the exception of ***Noise***, with its three ratings reasonably centered around a middle peak.

	Overall	food	ambiance	service	noise
overall	1.000	0.082	0.562	0.569	-0.032
food		1.000	0.077	0.086	-0.048
ambiance			1.000	0.543	-0.109
service				1.000	0.070
noise					1.000

Figure 2: *Pearson's R* coefficient s of correlation among ratings

As the correlation matrix in Figure 2 reveals, *Ambiance* and *Service* scores are substantially more correlated with *Overall* score than are *Food* and *Noise*, though all correlations shown have extremely high significance ($p < 0.0001$) given the huge number of correlated points (N=~450k).

Figure 3 provides a further illustration of these comparative correlations, using average values.

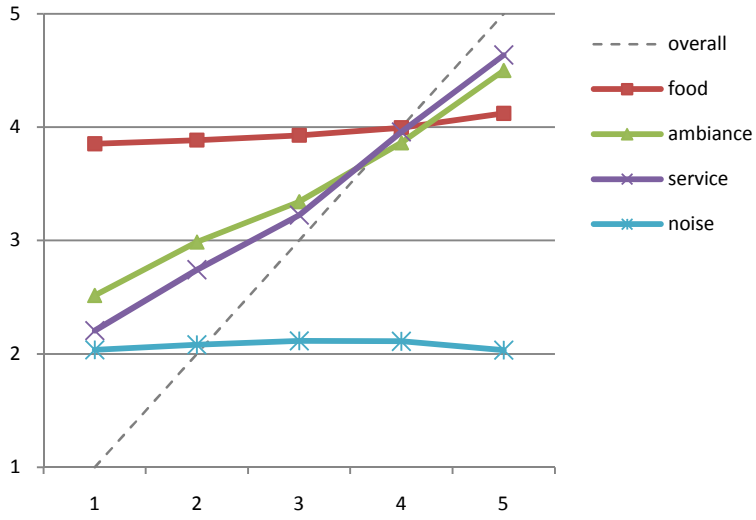


Figure 3: Average sub-topic rating for reviews with a given *Overall* rating. Dotted line represents perfect alignment with *Overall*.

While it seems intuitive that *Noise* may be somewhat dissociated from *Overall* score—consider that it depends upon the style of restaurant; pervasive quiet at a brew pub would probably be a significant negative—it's not immediately clear why *Food* should be less correlated with *Overall* than *Ambiance* and *Service*.

The answer likely depends on which is cause and which is effect. If it's the case that *Ambiance* and *Service* are driving *Overall* score—and to a greater degree than does *Food*—it may be that anything different from expectations (whether better or worse) on these highly subjective categories may stand out more in memory than *Food*, perhaps providing more fodder for post-hoc anecdotal recollections. In psychological terms, these elements would be considered more "salient." If cause and effect flow in the other direction, however, it may be that it is in fact *Food* where reviewers have more distinct memories, enabling a score more separable from *Overall*, where *Ambiance* and *Service* are largely just mirroring the *Overall* score. The latter explanation seems more likely to us, but this would be a good area for a follow-up interview study. Gaining such insight into the direction of causation might in turn enable us to better model these interactions in our SA system.

c. Scoring Curiosities

Finally, it's worth noting within this discussion the special challenge presented to an SA system by cases—and there are many—where there appears to be quite an inexplicable lack of alignment between what the reviewer has to say in

the text and the scores given. For each of the following examples, the reviewer provided an *Overall* rating of 5 (best), while our system guesses a 1—and here we find a 1 hard to argue with!

(4) *We were ignored from the moment we walked in. The couple that came in 3 min after us was seated first. We received terrible service from our server. My date was so upset he made a complaint and nothing was done about it. We will never eat again at the Crystal City location.*

(5) *The worst service I have received in a long time. I had to get up three times to go find our waiter. Never removed plates from our table. Never came back for drink orders. We were missing one dinner entree for 20 minutes. We would have totally stiffed the waiter which I've never done but they had a 18% mandatory gratuity charge for a party of 6. (…)*

Also consider the following, where each was rated a 1 by the reviewer, but our system guesses a 5, which again seems entirely reasonable!

(6) *What a great find. We had a wonderful time-the food and service was amazing. We will definitely be returning for more.*

(7) *Went for my husband's birthday. Great place for a special occasion. Service was impressive.*

Naturally, we might guess that these are essentially scoring mistakes—that these reviewers misunderstood and reversed the scale—but such speculation is of little help in training our system, as there is no getting around the fact that if a reviewer makes such a scale mistake, an SA system that "does the right thing" with the linguistic sentiment with which it's presented will inevitably score these "incorrectly."

3. EVALUATION

A significant design element for this project is the consideration of how best to evaluate results. As encouraged by Andrew, we focus on accuracy measures, but we do also consider several others. We report both *Training* and *Test* set accuracies, of course. We also, however, need to acknowledge that this isn't really a five-class unordered problem, but rather a 5-point scalar measure. So we introduce an *Offset* score, calculated as the average difference between actual and predicted. Finally, we also consider *Precision* and *Recall* ROCs.

4. CLASSIFICATION ENGINE

As previously mentioned, the project employs a Support Vector Machine classification framework. In particular, we make use of existing LIBSVM software.

5. FEATURE ENGINEERING

As in PA3, where we used a MEMM for Named Entity Recognition, here again attempting to devise clever features—with Andrew's Bag of Words baseline as the launching off point—is by far the largest part of the effort.

In prior projects we established a working process of iterative engineering—attempting a feature; evaluating the results; considering those results in designing a next feature. We do plenty of that here, as well, though in this case, there is also a body of prior work on SA with which it makes sense to "seed" our effort by simply immediately implementing a number of the features that others have found effective. To be clear, though, we never simply assume that they'll be helpful here, and in fact, a number of features suggested elsewhere proved ineffective within the present effort and the peculiarities of the given data.

After working through this available base of known ideas, we wade into further extensions and inventions of our own, as detailed below. We can further break this discussion into six man sections—***Preprocessing***, ***N-grams***, ***Black List*** (filtering) approaches, ***White List*** approaches, ***Topic Modeling***, and ***Entropy***-based—each of which will be elaborated below.

a. Basic Features

Once again, we began by implementing a large number of features used in prior SA work. In some cases these were fully our own implementations, our own spins on well-known features, while in some other cases we eagerly made use of available implementations from NLTK and other toolsets.

The resulting performance figures from this first set of features are summarized in Figure 4. These initial results were all generated on 20% of the full OpenTable corpus—91,481 reviews—further subdivided into a 10K-review training set, with ~80K reviews left as the test set. Line 7 in Figure 4 is an exception where we increased the training size to 80K reviews, with testing on the remaining ~10K left from our ~90K-review 20% sub-corpus.

		Accuracy		
Config #	**Shorthand model description**	**Train**	**Test**	**Avg Offset**
1	baseline (unigram / Bag of Words)	90.12%	50.13%	
2	+ pre-processing (contraction splitting; proper Noun, punctuation, and number stripping; lower casing)	69.60%	54.47%	0.540
3	+ basic features (POS filtering; negation processing; stemming)	56.29%	54.63%	0.617
4	+ bigram-only model	52.92%	52.40%	0.701
5	+ trigram-only model	49.96%	49.92%	0.765
6	+ mixed *n*-gram model	56.96%	54.80%	0.624
7	+ jumbo (80K reviews) training	54.53%	54.08%	0.611
8	+ "white list" (sentiment word list)	66.43%	56.59%	0.558
9	+ net sentiment words counter	66.58%	56.59%	0.557

Figure 4: Initial feature set and results

- **Preprocessing**
  - **Splitting**
    - **Contractions**
      A common technique in corpus linguistics, English contractions—both possessives, such as *Bill's*, and negations, such *isn't*, *weren't*, *hadn't*, and so on—are split into two tokens in the input stream. A contraction such as <*Bill's*> becomes <*Bill*> <*'s*>, while the <*n't*> of negation constructions is similarly tokenized separately.

    - **Punctuation**
      Similarly, all punctuation such as periods and commas were separately tokenized. They were not stripped entirely lest they prove useful for later classifier features, especially negation processing.

  - **Neutralization**
    - **Proper Nouns**
      Deemed unlikely to carry sentiment, these were converted to a single common <*PropNoun*> token before feeding our classifier feature.

    - **Numbers**
      Numbers were similarly neutralized, given their presumed lack of sentiment.

    - **Lower Case**
      Finally, all remaining text was lower-cased for later comparison operations.

The effects of such basic preprocessing alone (#1 in Figure 4) were significant, accounting for a 1.09x increase in *Test Accuracy*.

- **Part of Speech (POS) Filtering**

  On a suggestion from Dan Jurafsky, we first tried filtering in order to build features only on adjectives since these are generally the most sentiment-bearing words—in restaurant reviews, these are frequently elements such as *tasty, noisy, rude*, and so on. This was accomplished via use of the NLTK Unigram tagger. Higher-order *n*-gram tagging appeared to be prohibitively slow given our data size, and in fact simple unigram tagging seemed sufficient to ID most adjectives. We also experimented with using NLTK's Brill TBL tagger, but found no significant improvement in performance. Finally, initial success led to adding adverbs to the list as also potentially sentiment bearing, e.g., *hastily, sincerely*, and so on.

- **Negation Processing**

  We followed the course taken by Pang et al. 2002 in recognizing that the sentiment bearing of a word is generally inverted in the context of negation, such as following *not*. For example, *not bad* is actually a positive sentiment. As in the Pang et al. research, we handled this by prepending a *NOT_* marker to each word found following such a negation word and continuing until the next punctuation mark. We also extended the Pang et al. approach by similarly recognizing a slightly broader list of negation markers, including *no,* and *never*. For example, "We've *never* had good service there" shouldn't yield a positive sentiment from the word *good*, which this feature accomplishes by creating a separate classifier feature for sentiment word *NOT_good*.

- **Stemming**

  As with any classifier system, it is desirable to maximize the generality of features, and another common technique for accomplishing this is some form of stemming—the processing of stripping affixes (generally suffixes in English) in order to arrive at an approximation of the base lemma of a word. This facilitates equivalence comparison with other inflected forms of the same lemma. For example, we thus create a single more generalized classifier feature for all words that are commonly stemmed to *fast*, including *fast, faster,* and *fastest*. We implemented this approach using the NLTK Porter Stemmer routine, which implements a modified set of the well known fixed transformation rules laid out in Porter 1980.

Taken together (line #3 in Figure 4), these features produced only a very slight positive effect (1.003x) on *Accuracy*, which remains a point for further exploration in any follow work, as these common NLP processing techniques were expected to yield more. As in the examples in Negation Processing above, we see these features producing their desired effects in the post-processed token stream, but have no ready explanation for their lack of end-effect in classification accuracy.

- **N-grams**

  We attempted both bigram and trigram classifier features on top of the processed token streams described above, but these produced only negative effects in terms of *Test Accuracy*, as seen in lines 4 and 5 in Figure 4. Once again, these are very short reviews, and our analysis is that bigram and trigram features in this environment lack sufficient application—insufficient generality—to be effective on their own. However, our best performance comes when combining unigram, bigram, and trigram features. In our SVM implementation, this is accomplished by establishing fixed dictionary sizes per order of *n*-gram, limiting the number of *n*-gram features in the classifier to the most frequently applied such terms. Empirical testing to tune these parameters led to our best results found with each of these dictionaries set to size 1K. (See line 6 in Figure 4.)

- **"White List"**

  As an alternative to filtering the input stream based on POS, we tried another technique found successful in prior works: identifying sentiment words from a static list of known sentiment words. To do this, we downloaded and processed the sentiment word lexicon available from the Harvard General Inquirer project. This underwent the same preprocessing and stemming methods as the review text itself to maximize potential matches. This approach yielded further positive results, as seen on line 8 in Figure 4, a 1.05x

improvement in *Test Accuracy,* with a 0.91x yield (reduction, which is good!) in *Average Offset.*

- **Net Sentiment Words Counter**

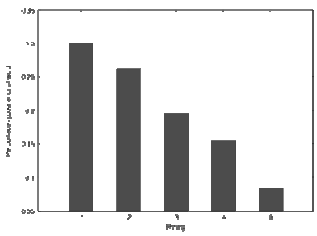
  As a final complement to the "White List" approach, we added—motivated once again by trying to understand why these features, while positive, were not getting still more traction—a classifier feature that presented a counter of the total number of sentiment words in a given review, where each positive word incremented the count, and each negative word decremented the count. As with all features, this was trialed both on its own and in various combinations, and we found that ultimately it produced a very small positive effect over and above the fundamental "White List" feature. (Line 9 in Figure 4)

As *Test Accuracy* figures are discussed in several places above, it should be noted that while we introduced our own measure of *Average Offset* in addition to provide a greater notion of scalar performance, rather than strict class accuracy, in practice we found—as seen throughout Figure 4—that these two figures generally tracked closely together (if inversely, since for *Accuracy,* higher is better; for *Offset*, lower is better) in terms of improvement (or lack thereof) through the introduction of additional features.

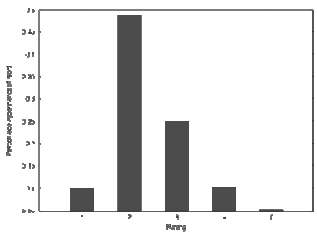
a. Strict Entropy Model

At this point in our project, we moved deeper into our data and error analysis in our attempts to move beyond the limited, if positive, results of our several features.

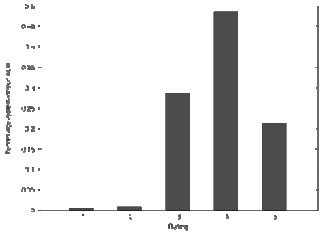
We decided to explore how the *n*-grams were distributed throughout the ratings. The results are interesting. We began by first examining the distribution of words across ratings, such that a word received five ratios (i.e., one for each rating). What we discovered was that certain words "described" certain ratings better than others. (Once again, this is not unlike the aforementioned findings by Chris Potts working with a different data set.) In the five graphs below, the word with the highest ratio of appearances for that rating versus the total appearances is compared with the ratios for the other ratings.



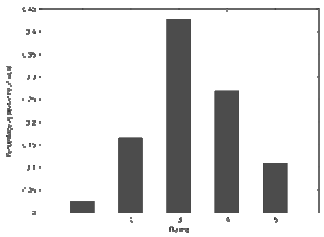
"worst"



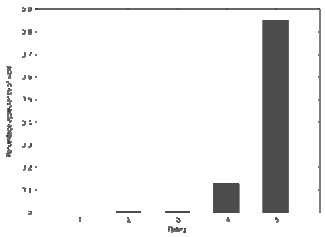
"mediocre"



"decent"



"solid"



"exceeded"

In addition, we generated the top words for each class, sorted by their appearance ratios, as listed in the following table:

1 Star	2 Stars	3 Stars	4 Stars	5 Stars
worst	mediocre	decent	solid	exceeded
awful	tasteless	ok	tight	exquisite
horrible	bland	undercooked	vibe	incredible
tasteless	overcooked	so-so	varied	impeccable
terrible	undercooked	okay	quieter	phenomenal

From these experiments, it became clear that there were certain words (and more importantly, certain *n*-grams) that appeared more often in specific rating classes. The hypothesis became such that if we could select features that had the largest skew, based on these distributions, we could develop a feature set that would easily separate the classes. Here we extended beyond the Potts analysis, as we understood it, to consider measuring the skew of the distribution by examining the Entropy of the distribution (i.e.,, the set of ratios). Thus, our model would use any of the aforementioned pre-processing techniques and/or feature selection techniques, and proceed to select the remaining features sorted by their Entropy. Specifically, the Entropy value *H(w)* for a single *n*-gram *w* is equal to:

$$H(w) = \sum_{i=1}^{5} \frac{r_{i,w}}{t_w} log_2(\frac{r_{i,w}}{t_w})$$

…where $r_{i,w}$ is the number of appearances of *w* in reviews with rating *i*, and t_w is the total number of appearances of the *n*-gram *w*.

We implemented this model, but unfortunately found accuracy slightly lower than the original Bag of Words model (56.4%). Thus, we needed to reevaluate our model and explore the data to see what could be going wrong. What we discovered was that the features being selected were primarily from the four- and five-star ratings. Why is this?

b. Frequency-Weighted Entropy Models

Due to the skew of the data toward primarily four- and five-star ratings, the ratio appearances for the words in these categories were much higher than those for the one- and two-star ratings. As indicated by the graphs above of the distribution of words among star ratings, there is a much higher disparity between "exceeded" for five-stars and other stars, as opposed to "worst" for the one-stars. One can think of it this way: if we have equal number of ratings in our data set, we may see "worst" 100 times for one-star ratings, and 10 times for the five-star ratings. On the other hand, if we have 10 times as many five-star ratings than one-star ratings, "worst" now appears 100 times for five-star ratings and thus equally for five-star and one-star ratings. This is clearly not indicative of the behavior of these classes.

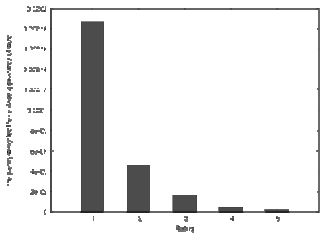
Thus, we decided to weight each of the ratios inversely to the frequency of each ratings class. In our updated model, we now divide the $r_{i,w}$ frequencies by the total number of ratings f_k for each class *k*. Thus, our new model is defined as:

$$H'(w) = \sum_{i=1}^{5} \frac{r_{i,w}}{Z t_w f_i} log_2(\frac{r_{i,w}}{Z t_w f_i})$$

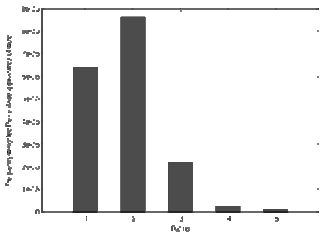
…where *Z* is simply a normalizing constant, defined as:

$$Z = \sum_{i=1}^{5} \frac{r_{i,w}}{l_w f_i}$$

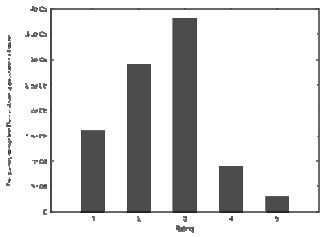
By using these ratios, as opposed to the original model, we now obtain more equally skewed distributions, as seen in the graphs below.



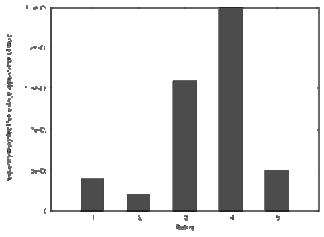
"worst"



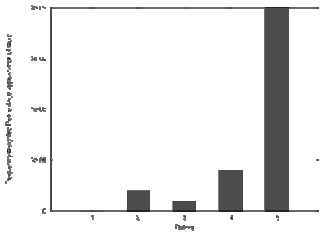
"mediocre"



"decent"



"solid"



"exceeded"

As seen in Figure 5 below—which ***begins*** by re-baselining with the new Entropy-based approach, then building again atop this new base—this model ultimately achieved our highest accuracies and was the most effective of all feature selection techniques we implemented, yielding a final improvement over the first baseline (line 1 in Figure 4 above) of 1.17x on *Test Accuracy*.

		Accuracy		
Config #	**Shorthand model description**	**Train**	**Test**	**Avg Offset**
1	Entropy unigram baseline (including pre-processing)	63.22%	57.13%	0.546
2	+ *n*-gram mix	63.22%	57.49%	0.553
3	+negation processing, stemming, net sent. words count)	64.36%	57.84%	0.542
4	+ stop words	64.29%	57.77%	0.544
5	+ POS filtering	55..55%	54.36%	0.618
6	+ sentiment word filtering ("white list")	62.24%	56.31%	0.581
7	+ increased *n*-gram dictionary sizes	71.73%	58.56%	0.517

Figure 5: Results using frequency-waited Entropy model

In this area, there should be much interesting future work, as there are perhaps more effective methods of re-weighting the distributions based on the ratings frequencies. In addition, it would be quite interesting to examine the distribution of other types of features, such as topic models, sentiment-only white lists, and so on. If, in fact, there exist distributions that skew in similar ways to these, these might also benefit from an Entropy-based model.

6. SUB-TOPIC MODELING

Finally, let's address performance on the scores other than *Overall*. In building our classifier implementation, we approached this in four steps.

a. Sub-Topic Classifiers

The first step was simply to take our best-performing model on *Overall* score—as noted above, this included the frequency-weighted Entropy mechanism in addition to earlier features such as negation processing, stemming, and so

on—then use it to train four additional classifiers on the other available scores. At this point, we aren't truly doing anything specific to topic; rather we're simply allowing the SVM to learn which *n*-grams are salient for a given score. Since, as previously discussed, very few reviews actually comment so thoroughly as to specifically touch on each sub-topic, it's not surprising that both performance on both *Test Accuracy* and *Average Offset* metrics for these unenhanced sub-topic tests is somewhat less than for the *Overall* score.

	Accuracy		
	Train	**Test**	**Avg Offset**
Food	49.80%	39.15%	0.876
Ambiance	60.71%	47.26%	0.628
Service	67.04%	53.70%	0.629
Noise	56.37%	48.43%	0.540

Figure 6: Sub-topic results using best-performing non-topic-enhanced general model

b. Topic Lexicon

The next step was to develop a list of topic-marking words for each of the available ratings. This was begun by culling by hand approximately 200 words from sampled reviews—for example, *menu, entree, dessert*, and so on for food; *waiter*, *manager, forget, greet*, and so on for service; etc. These lists were in turn fleshed out by accessing WordNet (via NLTK) to find synonyms (SynSets) for the initial items, which yielded an expanded set of lists totally ~500 topic words.

c. Topic-Filtered *N*-grams

The first of two features developed to employ this topic list information took the approach of filtering our previously produced *n*-grams by topic. Once again, we had a separate classifier for each sub-topic, so then within the classifier for a given topic, we now only build an *n*-gram classifier feature if the given *n*-gram included an item from the appropriate topic list. While this approach has the advantage—at least when using trigrams—of capturing cases where a sentiment word and topic word are not immediately collocated—for example <*soup*$_{\text{FOOD}}$ *was fantastic*$_{\text{ADJ}}$>—we did immediately anticipate the potential disadvantage that this would also capture potentially extraneous words in cases such as the alternative <*fantastic*$_{\text{ADJ}}$ *soup*$_{\text{FOOD}}$ *was*>, where the trailing copular verb probably does not enhance the effectiveness of the feature, working only to limit its generality. This led to the next feature, based instead on topic-word proximity.

d. Topic-Word Proximity Filter

This second topic-infused feature we developed in parallel—i.e., in this case, counter to much of our development process, we did develop both of these features a priori in order to test one against the other. Here, we focused on a unigram classifier model, but we "color" each word by topic based on its proximity to a known topic-list word. We then build sub-topic classifier features based only on unigrams found to be within *d* words of an item from the topic list appropriate for the given classifier, where *d* is a tunable parameter. Via experimentation, we set *d*=2. With this feature, for example, both the token sequence <*soup*$_{\text{FOOD}}$ *was fantastic*$_{\text{ADJ}}$> and <*fantastic*$_{\text{ADJ}}$ *soup*$_{\text{FOOD}}$ *was*> yield the same better-generalized feature <*fantastic*$_{\text{ADJ/FOOD}}$>.

Subsequent testing found the first feature, topic-filtered *n*-grams, to actually have a negative effect on accuracy. The latter feature, topic-word proximity filtering, yielded positive—if only very modest—results, ranging from 1.01x to 1.03x improvement in *Test Accuracy* for the sub-topic ratings, as seen in Figure 7. This remains an area where we would like to continue our investigations in the future.

	Accuracy		
	Train	Test	Avg Delta
Food	53.83%	40.05%	0.887
Ambiance	62.05%	47.88%	0.610
Service	63.78%	54.92%	0.621
Noise	51.33%	50.35%	0.505

Figure 7: Best and final scores for sub-topic ratings, using topic-word proximity filtering

7. CONCLUSIONS AND FUTURE WORK

Smart feature selection is the key to solving the Sentiment Analysis problem. While we found some success through the combination of previously worked features and our introduction of an Entropy-motivated feature, an obvious next step would be to explore different methods for capturing the context of words under topics, perhaps expanding on the topic models. In our experiments, we noticed that some of the top bigrams were phrases such as "good, but" and others that clearly act as a modifier for the text following it. In this way, smart analyses of these modifiers and building a model of sentiment, then modifying the succeeding text based on these modifiers would be an interesting experiment. In addition, the Entropy-based model certainly has hope and would benefit from further in-depth analysis into the weighting techniques, depending on the skew of the data. Furthermore, it would be important to capture differing distributions (i.e. right now a spike on a 1-rating, low on 2-rating is measured the same as low on 1-rating, spike on 2-rating), and attempt to get a uniform sampling of these different types. In this way, one could imagine applying a Kullback-Leibler divergence between different distributions and choosing those that maximize the difference. Of course, this would be an intractable problem, and thus would require interesting methods for optimization.

The data itself also needs work. As we mentioned, there are several examples of mislabeled data, which for the sake of analysis, should be removed from our data set. On the other hand, if one wished to develop a robust approach, one could develop a noise-resistant algorithm to better deal with poor training examples. In addition, some extra work in scrubbing the data of "impossible" data instances would be instructive (such as those with less than five words). A more ambitious next step would be to run a study—perhaps using a crowd-sourced approach such as Mechanical Turk—to ask humans to guess at ratings given the review text, then measure the differences between what a human would recognize and what the actual rating values are. We suspect that there would be significant variation between the labeled and MTurk re-labeled data. This would give some insight into how possible it could be for our algorithm to classify these instances well.

Finally, we restricted ourselves to an SVM training method for the purposes of exploring NLP-based methodology, rather than general machine learning. Perhaps other learning algorithms could provide better accuracy and offset values in our follow-on work.

8. PROJECT TEAM

Robin Melnick – Research, basic features / preprocessing, POS, sentiment lexicon, WordNet, topic modeling lexicon, data and error analysis, write-up.

Dan Preston – Core architecture, myriad features, topic modeling, Entropy-based scheme and implementation, data and error analysis, write-up

9. REFERENCES

Pang, B. and L. Lee. 2004. A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts. *Proceedings of ACL*: 271--278.

Pang, B., L. Lee, and S. Vaithyanathan. 2002. Thumbs up? Sentiment classification using machine learning techniques. *Proceedings of the Conference on Empirical Methods in Natural Language Processing (EMNLP)*: 79--86.

Porter, M.F. 1980. An algorithm for suffix stripping. *Program*, 14(3): 130−137.

Snyder, B. 2007. Multiple aspect ranking using the Good Grief algorithm. *NAACL.*