User Review Sentiment Classification and Aggregation

Steven Garcia
garcias@stanford.edu

Ping Yin
pingyin@stanford.edu

ABSTRACT

User reviews provide a wealth of information but are often overwhelming in volume. In this work we propose a novel approach to extract positive and negative sentiments from user review data leveraging only the overall review scores that are part of the data itself. We then investigate clustering techniques to identify key positive and negative sentiment aspects to provide a user friendly summary for users.

1. INTRODUCTION

User reviews on product websites such as Yelp or Amazon provide a wealth of information about the subject of the review. In many cases when selecting a service (i.e. restaurant) or when choosing a product (i.e. book, dishwasher etc.) the amount of information available in user reviews can be of greater volume and more trustworthy than the official product descriptions provided by the vendor.

The number of reviews for a single item can at times be overwhelming. For instance, the Amazon Kindle keyboard 6" e-reader has over 42,000 user reviews as of this writing [1]. The review reader can gain an understanding of overall satisfaction by looking at summary statistics such as the average rating or score, but the details about what make each product great or not are hidden inside the body of the reviews themselves. With such a large amount of data to process, it would be beneficial to automate the process and generate a summary of the reviews.

In this project we propose an approach to review summarization that leverages trained models for classification and clustering to derive sentiment for the key aspects of a reviewed item. Given a set of item reviews, our system will output sentences from the reviews that are representative of key aspects for which users had strong positive and negative opinions about.

This paper is structured as follows. In Section 2 we discuss related works in the ares of sentiment detection and analysis. In Section 3 we provide an overview of the Amazon and Yelp datasets used in our experiments. In Section 4 and Section 5 we outline our novel approach to review summarization and present results. Finally, in Section 6 we conclude our findings and outline future directions of exploration.

2. BACKGROUND

Liu and Zhang [7] outline five tasks that are core elements of sentiment analysis: entity extraction and grouping; aspect extraction and grouping; opinion holder and time extraction; aspect sentiment classification; and opinion quintuple generation. In this work we will focus on the tasks of aspect extraction and grouping, and aspect sentiment classification.

Pang et al. [10] successfully use bag-of-words features to predict positive and negative labels for movie reviews. They find little difference between Naive Bayes and SVM classifiers. Based on this work our experiments begin with bag-of-words features and Naive Bayes classifiers.

Das et al. [5] classify user sentiment from user message boards using a range of algorithms and show that model selection can have a significant impact on accuracy. Similarly, we experiment with several models for our classifier and choose our best model using precision and recall metrics.

Cataldi et al. [4] propose a system that leverages NLP to identify feature level sentiment. There is significant cost in parsing sentence structure and typically such works is limited to small data sets. Although we utilize large data sets in this work, we explore the the impact of POS tagging in a later component of our system where the cost to generate this data is reduced.

3. DATASET

In this work we experiment with two datasets. The Yelp Challenge dataset containing 1.6M reviews spanning 61K businesses and 366K users [11]. We also utilize the Amazon product data set which provides nearly two decades of reviews for a variety of product segments [8]. From the Amazon dataset we experiment with the Video Games, Tool and Home Improvement, and Musical Instruments subsets. For all datasets, each record contains a review text and an overall review score. This work is limited to consuming those two fields but could be extended to include the data provided other fields.

For each review we breakdown the reviews into sentence snippets and then into tokens to use as features. Additionally some experiments utilize stemming and part of speech tagging. All text manipulation, stemming and tagging was performed using the Natural Language Toolkit [2]. Figure 1 shows the distribution of sentences per review and the breakdown of review scores per dataset.

4. METHODS

Our goal is to identify sentences that are representative of positive and negative key aspects of the review subjects. To this end we must establish a mechanism to identify positive and negative sentiment within the body of each review.

A key challenge to classifying review sentences is that we do not have labels for each individual sentence within the data set. Given the large number of reviews in our data sets we instead label each sentence with the overall review score. The overall review score therefore acts as a surrogate score for each sentence in that review. Although noisy, we can justify this decision for two reasons. First, due to the overwhelmingly large amount of data we expect that the learning algorithm will successfully learn to classify sentences to a reasonable degree. As shown in Figure 1, all but one of our data sets has at least 100,000 reviews at every rating. Second, our extraction of representative aspect sentences does not require every sentence to be correctly classified. Indeed, even with a large number of unclassified sentences, we can still produce a meaningful summary for the user. In Section 5 we measure the precision of this approach.

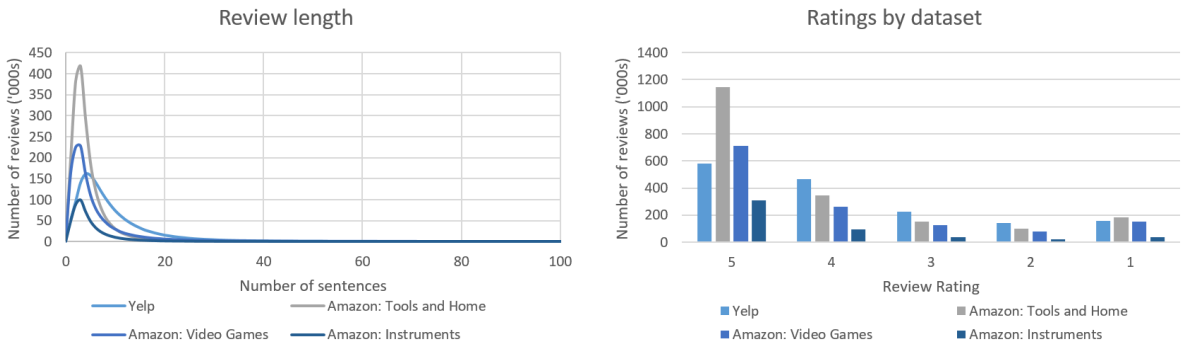

Figure 1: Dataset review and length breakdown

Having established a classification of positive and negative sentences within a review, we can then apply a clustering algorithm to find groupings which are representative of key elements.

An overview of our system is depicted in Figure 2. The Sentence Parser splits reviews into sentence fragments. The Sentence Classifer then assigns a *positive*, *negative*, or *unsure* label to each sentence. The Clustering component groups together common sentences of each labeled class. Finally, Aspect Selection filters the clusters and outputs a representative sentence for each selected grouping.

4.1 Classification

To classify the sentences as positive or negative we consider three candidate algorithms. Naive Bayes classifiers are well explored in the literature for use on text classification problems. They are a natural fit for text classification as they are a generative class of models that estimate the likelihood of observing the data. Specifically, the classifier uses Bayes rule to determine the likelihood of a class given the data with $P(y|x) = \frac{P(x|y)P(y)}{P(x)}$, and using an independence assumption this reduces to $\frac{(\prod_{i=1}^{n} P(x_i|y=j))P(y=j)}{\sum_{k}^{K}(\prod_{i=1}^{n} P(x_i|y=k))P(y=k)}$ for a set of features n, aiming to predict the likelihood of observing class j in the set of classes K.

McCallum et al. [9] compare Baysian models and find that the Multinomial Naive Bayes classifier is almost uniformly better than the Bernoulli classifier. However in this work we deal with very relatively short documents (individual sentences). The Multinomial model specifically accounts for feature frequency in determining the likelihood of a document. This makes sense for longer documents, but for short sentences (like the ones we aim to classify) it could be argued that the frequency of the features is less important than the actual occurrence of those features. As such we consider both the Bernoulli Naive Bayes classifier and the Multinomial Naive Bayes classifier.

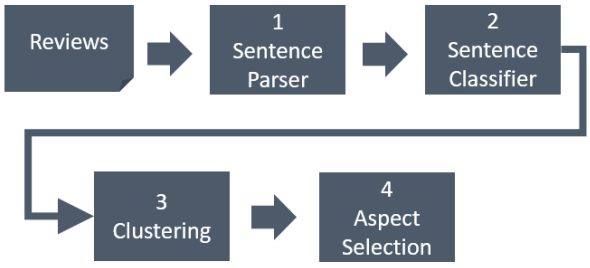

Figure 2: Overall system architecture

One limitation of the Naive Bayes classifiers is that they make an independence assumption. A tree classifier can implicitly capture dependency between features. To explore this we also experiment with a gradient boosted tree classifier [3]. This classifier is built as the aggregation of a collection of trees. To classify an instance, the set of trees are evaluated and the average score of all trees is taken as the classification result.

To train the model, on each training iteration a new tree is constructed to maximize correct classification of the training set. To build each tree, nodes are added by selecting the feature that best partitions the data at the current node into the target classes. The process is repeated for the output of each node up to a maximum depth which is specified as a parameter. One optimization to reduce training time is to randomly sample features for each node partition (as opposed to trying every possible feature). A related optimization is to sample from the training data for each training iteration (bag-fraction). Both of these optimization are controlled through parameters. To avoid over fitting you can specify a minimum number of samples in a leaf node. Other parameters used by the model include the learning rate, and the maximum number of trees to train.

An interesting property of this algorithm is that as each tree is added, the learning algorithm will evaluate the accuracy of the model and apply a weight to each training sample such that the next tree is biased towards correctly classifying those training samples that were previously classified incorrectly. In this way, each new tree is more targeted towards those subsets of the training data that are hard to classify.

As noted in Section 4, our labels are approximated for each input sentence using the overall review score. To reduce ambiguity for our model we train on only very negative (one star) and very positive (five star) reviews. In Section 5 we show that this results in a classifier that is suitable for the task.

Our features are tokens (or terms) extracted from the review sentences. For each token we count the number of occurrences of that token in the training sample. We choose to stem our tokens to allow the model to better leverage the features. Stemming is applied using the NLTK library. To reduce the feature space we stop terms that occur in less than 1% of our training data with the expectation that such rare terms will not impact the final clustering result.

The final step for deploying our classifier is to ensure high accuracy. As not every sentence can be classified successfully as positive or negative, we apply two thresholds to our

classifier. A high threshold is used to limit which instances are classified as positive, and a low threshold is used to limit the negative classifications. Anything in between is considered *unsure*. Using this approach we can control for any target level of precision on the classification. We select a target of 90% precision in this work, although this can be tuned as desired with a trade-off on recall.

4.2 Clustering and Aspect Selection

With a trained classifier, we now consider the task of clustering and aspect selection. We use a K-Means [6] classifier to create clusters of similar sentences among the positive and negatively classified results. K-means works by establishing k centroids in an n-dimensional space, and then for every data point assigning that point to the nearest k-centroid. Once complete the centroids are updated to be the actual center of all items assigned to its cluster. The algorithm then iterates the above steps until the centroids have converged. Distance is typically the Euclidian distance as given by $\sum_{i=0}^{k} \sum_{x \in S_i} ||x - \mu_i||^2$, where S is the set of data points to cluster.

One challenge is that the number of clusters within a positive or negative sentence set is unknown before applying the clustering algorithm. To address this issue we propose a criteria for valid clusters and iterate the clustering algorithm over a range of k values. Once complete we select the value of k that resulted in the largest number of valid clusters. We define a valid cluster to be one where the most frequent 5 features/terms can be found in more than 90% of the cluster, and where the cluster consists of 10 or more different review sentences. The first criteria is a heuristic which ensures that the cluster contains a subset of terms that will be identifiable for the aspect. The second criteria ensures that our cluster is representative of multiple reviews. The parameters are configurable and explored in Section 5.

Finally, having clustered a set of classified sentences, we select a representative example from each valid cluster, by selecting the sentence that is closest to the centroid using Euclidiean distance.

5. EXPERIMENTS

For all experiments below we sample data as specified in the experiment description. The sampled data is then split into training and test sets at a ratio of 70% train and 30% test. Results presented are for test data unless otherwise specified.

Our first experiment compares Naive Bayes classifiers with the gradient boosted tree classifier for the Yelp data set. Not every sentence will be strictly positive or negative (indeed some may be both). As such we aim to classify only the strongly positive and negative sentences for our final results. To achieve high precision we apply two thresholds, one on the upper end for positive sentiment, and one on the lower end for negative sentiment. Figure 3 shows the recall/precision curves on the test data as we vary thresholds for both positive and negative classifiers. Models were trained using 200,000 positive and negative training samples. The results show that the gradient boosted tree provides superior accuracy on almost all points of the curve for classifying sentences back to the original label. Based on this result we select the gradient boosted tree classifier for further experiments.

As discussed in Section 4.1, the gradient boosted tree model requires various parameters. To get the best performance out of the model we run a sweep across the parameter space selecting 100 possible parameter combinations and choose the best performing model. Figure 4 shows the how tuning the parameters can improve the performance of the model by showing the reduction in L2 error. Each time we trained our model, we ran the parameter sweep. We found our best model (including the improvements discussed in the remainder of Section 5) using 1,000 trees, a learning rate of 0.17, a bag fraction parameter of 0.63, feature sampling of 0.46, 200 minimum labels per leaf, and a maximum tree size of 264. Finally we select our thresholds to obtain 90% precision for both the positive and negative classifiers.

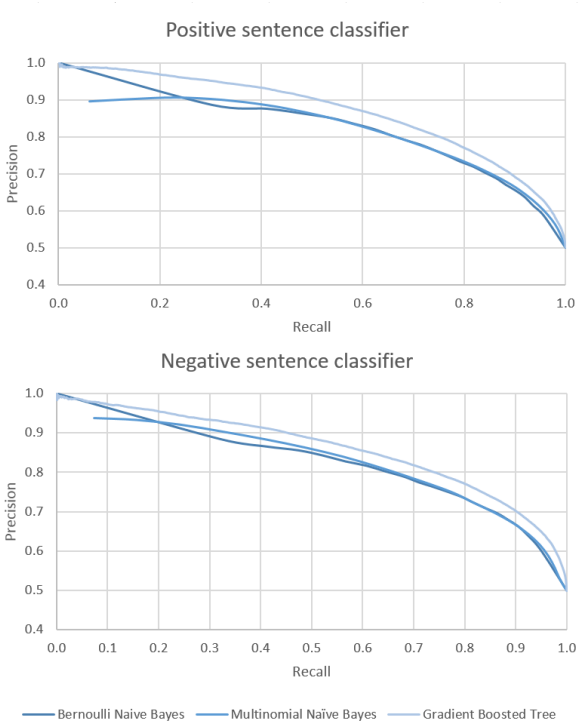

Figure 3: Recall/precision curves of positive and negative classifiers for candidate algorithms

One variable that can impact the performance of a model is the number of training samples to use. To explore this we trained our model using a range of training set sizes. Figure 5 shows the impact on recall for the positive and negative classifiers when targeting 90% precision. Note that after splitting the Yelp data set reviews into sentences we

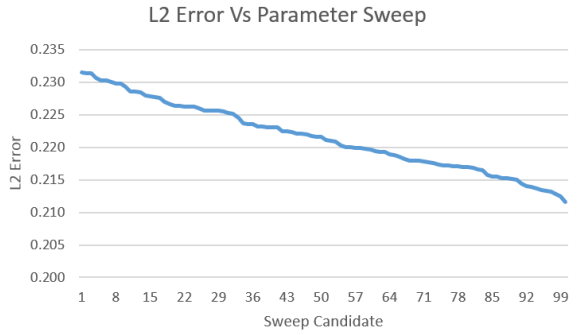

Figure 4: L2 error on test set for gradient boosted tree over model parameter space for Yelp data set

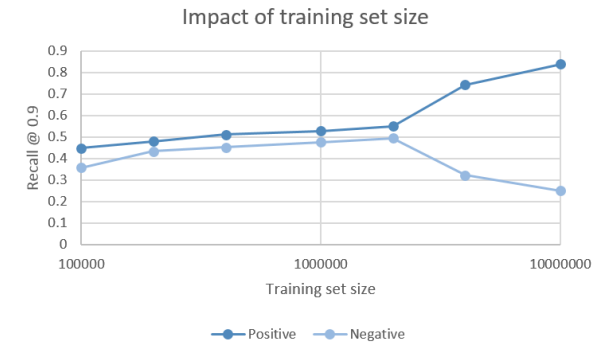

Figure 5: Effect of training set size on recall at 90% precision

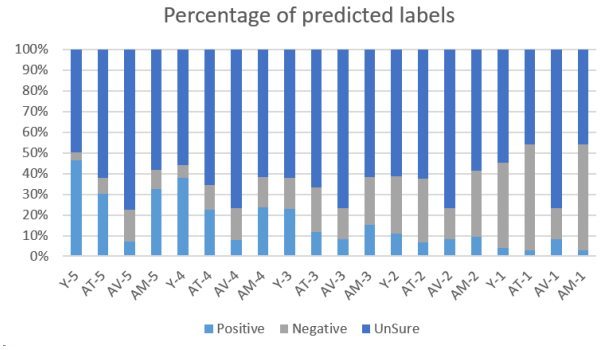

Figure 6: Distribution of predictions across labels for all data sets after training model. Yelp (Y), Amazon: Tools and Home (AT), Amazon: Video Games (AV), and Amazon: Musical Instruments (AM)

have on the order of 14 million sentences overall and just over 2 million 1-star sentences. The figure shows that once we are unable to select a balanced amount of positive and negative training samples, our classifier begins to bias towards a positive prediction. It is worth noting that before we reach the point of imbalance we see that recall continues to increase as we add more training data. This suggests that obtaining more data could help, however the rate of improvement is diminishing so it is unclear how much more recall we could obtain for the cost. We chose to train our final model using 2 million positive and negative samples.

Our model is trained using sentence words as features. A problem with this approach is that terms like *rose* and *roses* are considered completely independent features although they represent the same aspect. By applying stemming to our feature set we can combine these features and allow the model to better leverage the signal provided by these features. At our selected precision level of 90%, recall for the Yelp positive classifier increases from 46% to 51%, and for the negative classifier it increases from 40% to 45%. Before stemming our model uses 2,195 features, and after stemming it uses 684 features.

Having trained and tested our model on the one and five star data, we now assess the performance of the model for all candidate sentences. Figure 6 shows the distribution of predicted labels when compared to the overall review label for our data sets. As expected a large fraction of the sentences are classified as *unsure*. For all data sets there is a clear trend of predicting a larger fraction of positive cases for reviews that are labeled positive (4 and 5 stars) when compared to classifying negative cases for the negatively labeled reviews (1 and 2 stars).

A critical assumption in this work is that the overall review label can act as a surrogate for the sentence label. To verify that this assumption we randomly sampled 400 classified sentences from 2 entities in the Yelp data and manually assigned labels of positive/negative/neither to those sentences. Table 1 shows that the positive predictor achieves our expected precision of 90%, classifying only 19 *neither* labeled examples as positive. The negative classifier is less accurate than expected with only 60% true negatives in the negative prediction set and 14% positives classified as negative. We discuss potential improvements for the negative classifier in Section 6.

With classified sets of positive and negative sentences for each review subject we now apply clustering and select representative sentences from clusters. We run k-means clustering with a range of values for k from 10 to 100 and observe three general patterns in the results: first, a cluster may be too *narrow* with a handful of sentences; second a cluster may be too *broad*, this often seemed to be the case for a

Table 1: Accuracy of predictions

	Prediction	
Judgment	Positive	Negative
Positive	181	28
Neither	19	63
Negative	0	119

Table 2: Cluster filtering accuracy

K	Narrow	Broad	Useful
10	7	1	0
20	14	1	0
30	21	1	0
40	29	1	0
50	31	1	0
60	44	1	0
70	52	1	0
80	61	1	0
90	71	1	0
100	85	1	0

Table 3: Cluster refinement with noun tokens

All features		Noun features	
Topic	**Useful**	**Topic**	**Useful**
attentive staff	Y	beef wellington	Y
crab cake	Y	broad menu	Y
chocolate strawberries	Y	chocolate stawberries	Y
filet mignon	Y	classy	Y
food dressings	Y	sommelier	Y
reservation	Y	sorbet	Y
romantic setting	Y	steak	Y
rose flowers	Y	table side salad	Y
salad cart	Y	bottle	N
seafood	Y	course	N
sommeli	Y	experience	N
table side food	Y	house	N
fantastic	N	old	N
four queen (location)	N	restaurant	N
incredible	N	thing	N
red	N		
share	N		
treat	N		

Table 4: Examples from final results

Data set	Item/Business	+/−	Aspect Representation
Yelp	Hugo's Cellar	+	Wonderful Caesar **salad prepared table side** to our specifications.
		+	They also have an extensive **wine** list, and a knowledgeable **sommelier**.
		−	Tucked underneath the **Four Queens Hotel** in **downtown Las Vegas**, this place...
Amazon: Tools and Home	G7 Power Minden LED Recessed Bulb	+	Got these to replace **cfl bulbs** in my great room and kitchen and all I can say is hot damn I love these things.
		+	No more **dimmer** problems and saves me quite a bit of money on my electric bill.
		−	I'm going to just eat the rest of them as the **burn out**, and replace them with a known brand.
Amazon: Video Games	God of War: Ascension	+	I particularly found the **Blades** of **Chaos** system to be the best it has ever been in the series.
		+	The **puzzles** in God of War are great, sometimes exceedingly challenging (especially in the second game)...
		−	**Ascension** just feels like a basic bare bones God of War without all the memorable moments...
Amazon: Instruments	Brokeback Mountain	+	Both **Michelle Williams** and Anne Hathaway give commendable performances as....
		+	**Simply** put, this is the best **movie** I've seen in a while.
		−	I thought this was supposed to be a love **story**??

single cluster which acted as a catch all for the sentences that the algorithm was unable to classify more specifically; finally there were clusters for groups that tended to represent a small group of features with a moderate number of sentences. In an effort to select sentences that are of value to the user we propose a filtering criteria for selecting clusters which removes a cluster that consists of less than 10 review sentences. This restriction allows us to select only aspects that are representative of a wide number of reviewers. Secondly we require that a large portion of the cluster (we select 90%) be represented by at most 5 features/terms. That is, a subset of 5 features must be present in 95% of the sentences belonging to the cluster. This allows us to ensure that a cluster has a small focus/aspect as represented by the feature set.

To examine our filtering criteria we selected a sample review set, applied our clustering algorithm and filtering criteria. We then labeled the filtered clusters for each value of k. Table 2 shows for each value of k the number for removed clusters in each identified class. The results show that no clusters are removed that would be useful for the user.

With confidence that our cluster filtering criteria can remove clusters of little value to the user we now examine the clusters that successfully pass through the filter. After clustering and filtering we observed some clusters are still generic in nature, for example a cluster formed primarily around the feature *incredible* can contain many aspects. To address this issue we applied POS tagging and attempted to cluster only features that were also classified as nouns. Table 3 shows how restricting clusters to noun features changes the types of clusters formed. For each cluster we provide a manual label summary to describe it, and a label indicating if it would provide useful information to the user. However there are still clusters that are centered on concepts of little value to the user such as *house* which can relate to *house wine* or *on the house* and so on. Further, applying this restriction led to a loss of aspects that were useful such as *attentive staff* and *reservations*. We proceed without POS tagging and note that we need to further refine the clustering selection in future work.

Finally to select results for the user we choose the sentence that closest to the center of the cluster. We present a sample of the final results for our system over the various data sets in Table 4. We can see that the negative clusters lack precision in some cases due to the lack of accuracy at the classification phase (as observed in the Yelp and Video game examples). Similarly there are cases where the positive cluster has an aspect which is of questionable value (for example the *simply* cluster for the instrument data). However, overall the system is able to identify clusters pivoted on aspects that users should find of interest.

6. CONCLUSIONS AND FUTURE WORK

We have developed a system to process a large number of reviews and report sentences from the reviews that are representative of the best and worst aspects. We found that with only review level labels, we could adequately train a classifier to predict positive sentences. For classification we recommend a gradient boosted tree over Naive Bayes classifiers. We developed a heuristic to filter the number of clusters to present to the user for each classified review set. Finally we show that that our system correctly identifies key aspects with end-to-end examples across multiple data sets.

Our approach of utilizing the review score for sentence level labels worked well for positive sentences, but was less effective for negative sentences. Labeled data sets at the sentence level may provide richer data with which to train our model. Use of POS tagging was unable to improve the performance of our clustering algorithm, however the literature suggests this is a candidate avenue of exploration for the classifier. Our clustering at times formed clusters around common concepts. Advanced NLP techniques may be of value to help improve the clustering component of our system. Finally, our experiments made some decisions with respect to parameterization, feature space and training data that although pragmatic are open to scrutiny. We intend to explore these decisions along with an exploration of other mechanisms for selecting the optimal number of clusters as described in the literature.

7. REFERENCES

[1] Amazon. Kindle keyboard, wi-fi, 6" e ink display. `http://www.amazon.com/Kindle-Wireless-Reader-Wifi-Graphite/product-reviews/B002Y27P3M/`. Accessed November 11, 2015.

[2] S. Bird. Nltk: the natural language toolkit. In *Proceedings of the COLING/ACL on Interactive presentation sessions*, pages 69–72. Association for Computational Linguistics, 2006.

[3] C. J. Burges. From ranknet to lambdarank to lambdamart: An overview. Technical Report MSR-TR-2010-82, June 2010.

[4] M. Cataldi, A. Ballatore, I. Tiddi, and M.-A. Aufaure. Good location, terrible food: detecting feature sentiment in user-generated reviews. *Social Network Analysis and Mining*, 3(4):1149–1163, 2013.

[5] S. R. Das and M. Y. Chen. Yahoo! for amazon: Sentiment extraction from small talk on the web. *Management Science*, 53(9):1375–1388, 2007.

[6] J. A. Hartigan and M. A. Wong. Algorithm as 136: A k-means clustering algorithm. *Applied statistics*, pages 100–108, 1979.

[7] B. Liu and L. Zhang. A survey of opinion mining and sentiment analysis. In C. C. Aggarwal and C. Zhai, editors, *Mining Text Data*, pages 415–463. Springer US, 2012.

[8] J. J. McAuley, C. Targett, Q. Shi, and A. van den Hengel. Image-based recommendations on styles and substitutes. In R. A. Baeza-Yates, M. Lalmas, A. Moffat, and B. A. Ribeiro-Neto, editors, *SIGIR*, pages 43–52. ACM, 2015.

[9] A. McCallum, K. Nigam, et al. A comparison of event models for naive bayes text classification. In *AAAI-98 workshop on learning for text categorization*, volume 752, pages 41–48, 1998.

[10] B. Pang, L. Lee, and S. Vaithyanathan. Thumbs up?: sentiment classification using machine learning techniques. In *Proceedings of the ACL-02 conference on Empirical methods in natural language processing-Volume 10*, pages 79–86. Association for Computational Linguistics, 2002.

[11] Yelp. Yelp challenge dataset. `https://www.yelp.com/dataset_challenge/dataset`. Accessed October 16, 2015.