Extracting and Aggregating Aspect-Level Sentiment from Product Reviews

Matthew Long, Desmond C. Ong, Shane Soh
Stanford University
{mlong14, dco, shanesoh}@stanford.edu

Abstract

Previous work in *aspect-level sentiment analysis*—identifying the sentiment associated with products and their individual attributes, like battery life—have mostly been formulated as supervised learning problems, requiring known labels of both the relevant aspects and their sentiment. Here we propose a hybrid method where we first generate aspects from natural language text via unsupervised clustering of word vector representations, and secondly extract aspect-level sentiment. We then proceed to summarize these aspect-level sentiment within and across reviews and provide visualizations to aid consumers.

1 Introduction

In today's e-commercialized society, the consumer not only has access to online stores through which she might purchase anything she desires, but she also has unprecedented access to a deluge of information—most notably, product reviews written by other consumers—with which she can make her decision. Unfortunately, sifting through hundreds of reviews across tens of different websites to acquire specific information about the product and its important attributes (e.g. *battery life* for electronics) is a time-consuming chore. It is difficult to automatically summarize this information, especially because of the heterogeneity of attributes across different product categories. Because of this, there has been much recent research tackling the two separate but connected components of this problem: (1) automatic discovery of aspects, and (2) aspect-specific sentiment analysis.

Within the context of a product review, the first component, *aspect discovery*, involves identifying individual "aspects" (which could be the product itself, or features/attributes of the product). This is made more difficult by the fact that relevant aspects might differ across similar products. For example, for a given electronic item such as a mobile phone, relevant aspects could be the battery life, screen size, weight, cost, and more. Not all these aspects are relevant across other electronic items: for a computer, weight might not be an issue, and screen size might be moot for headphones. Previous methods have relied on graphical models (e.g. Latent Dirichlet Allocation [1-3], or Conditional Random Fields [4]) to model latent aspects. We propose using similar unsupervised methods to generate candidate aspects via semantic word vector representations.

The second component, *aspect-specific sentiment analysis* involves subsequently identifying the sentiment—positive, neutral, or negative judgments—associated with aspects [1-6]. In this paper, we propose extending recent successful advances in deep learning [6-7] to address this problem. In particular, we seek to improve and extend the work in [6], who used deep learning architectures like the Recursive Neural Tensor Network (RNTN) to extract aspect and sentiment in a single step. Unfortunately, supervised methods (e.g. [5-7]) that require labeled aspects-sentiment pairs for training are unscalable, and a better approach would be to automatically identify relevant aspects for each product (which, e.g. [1-3] try to do via topic modeling).

2 Problem Statement

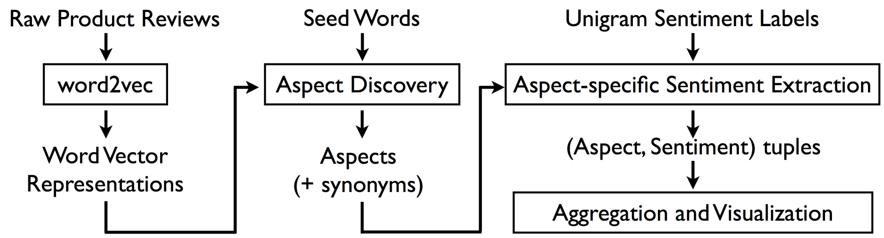

Figure 1: Proposed Workflow. Boxed items are processes, while non-boxed items are outputs. The left third of the diagram involves converting the reviews into a word vector representation, the middle third details unsupervised aspect discovery, while the right half of the diagram involves aspect-specific sentiment extraction, summarization and visualization.

Our proposed workflow can be divided into two main parts: (unsupervised) **Aspect Discovery** and **Aspect-Specific Sentiment Extraction**. See Fig. 1 for an illustration.

2.1 Aspect Discovery

Problem: Given a short list of attributes ("seed" attributes), we discover an expanded list of attributes by returning the top n related word vectors based on cosine similarity.

Dataset: 6 million reviews of electronic products from Amazon.com [8].

2.2 Aspect-Specific Sentiment Extraction

Problem: Given a set of product reviews (for a single product), identify (aspect, aspect-related sentiment) tuples. This is a similar problem statement as [6]: we intend to expand upon their model.

Evaluation: We will construct word cloud visualizations color coded to express sentiment. We will also perform comparisons with "expert review" sites like CNET and DPReview (e.g. DPReview specifically rates digital cameras along certain chosen dimensions).

3 Technical Approach and Models

3.1 Aspect Discovery

The first part of the project involves an unsupervised approach to generating aspects. We tokenized our corpus using NLTK's punkt tokenizer [9] for sentence splitting. We then removed all non-alphanumerical characters and replaced all digits with `DG`. Finally, we performed collocation detection to detect common bigrams. We run word2vec (CBOW, Skipgram) to obtain a word vector representation of the data (varying hyper parameters like window size). We start with a list of seed attributes across a wide range of products, and discover more via closest cosine similarity.

3.2 Aspect-Specific Sentiment Extraction

An additional step would be to label all unigrams with a sentiment analysis lexicon (Bing Liu's sentiment lexicon) or the Sentiment pipeline in CoreNLP [7]. With the aspect-labels from the previous part, we would have a dataset labeled with aspect and (unigram-) sentiment. Next, we build off and improve on the model in [6]: we will use a RNTN or improved architecture to extract (aspect, aspect-specific sentiment) tuples from the data. Finally, we will have a summarization of aspect-related sentiment via visualization of our output, in terms of a word cloud of important aspects (e.g., sized by word frequency and colored by sentiment).

4 Intermediate/Preliminary Experiments & Results

4.1 Model Training

We trained three different word2vec models. The most successful model was trained using the CBOW (continuous bag-of-words) model with a window size of 10 and feature dimension size of 300. We also ignored all words with total frequency count below 40 (this helped to remove many typos). We determined the performance of each model by querying the model with various aspects common to electronic products (e.g. "portability", "screen quality", etc).

4.2 word2vec Results

We queried our word2vec model and returned the top-10 results based on cosine similarity of the word vectors.

Query	portability	contrast	tripod
Results	(u'portability,', 0.7285999655723571 8), (u'compactness', 0.64743077754974365), (u'mobility', 0.60842603445053101), (u'versatility', 0.5763777494430542), (u'simplicity', 0.53962129354476929), (u'lightness', 0.53950369358062744), (u'convenience', 0.53897607326507568), (u'ruggedness', 0.5272858738899231), (u'versatility,', 0.5055851936340332), (u'thinness', 0.49253776669502258)	(u'contrast,', 0.65686732530593872), (u'sharpness', 0.62712550163269043), (u'color_saturation', 0.60933655500411987), (u'saturation', 0.57076853513717651), (u'brightness', 0.5553707480430603), (u'gamma', 0.53090476989746094), (u'shadow_detail', 0.52805298566818237), (u'color_accuracy', 0.52408510446548462), (u'dynamic_range', 0.52167940139770508), (u'black_levels', 0.51741272211074829)	(u'monopod', 0.74430769681930542), (u'tripod,', 0.71975594758987427), (u'ball_head', 0.70861411094665527), (u'tripods', 0.68399727344512939), (u'ballhead', 0.60356354713439941), (u'manfrotto', 0.59598124027252197), (u'monopod,', 0.58229464292526245), (u'pole', 0.56997144222259521), (u'quick_release', 0.549965500831604), (u'cold_shoe', 0.5460544228553772)

Table 1: Preliminary Results from word2vec model

As shown in the table above, words like `portability` returned many synonyms as well as product aspects that are related to it (e.g. `ruggedness` and `simplicity`). We also find that our model is capable of returning aspects that are specific and unique to the product category it is trained on. In this case of electronic products, a query like `contrast` returned words like `shadow_detail`, `dynamic_range`, `black_levels` and `gamma`, which are aspects specific to devices like monitor displays and cameras.

A query like `tripod` returned various aspects of camera tripods, many of them being features that are non-obvious to the layperson. For instance, `ball_head` refers to a ball-type tripod heads and `quick_release` refers to quick-release tripod mounts. Many of these queries would otherwise perform poorly if we were to use lexical databases like WordNet.

Acknowledgments

We would like to acknowledge funding for computing resources provided by the Deep Social Learning Lab at Stanford.

References

[1] Titov, I., & McDonald, R. T. (2008). A Joint Model of Text and Aspect Ratings for Sentiment Summarization. In *ACL* (Vol. 8, pp. 308-316).

[2] Jo, Y., & Oh, A. H. (2011). Aspect and sentiment unification model for online review analysis. In Proceedings of the fourth ACM international conference on Web search and data mining (pp. 815-824). ACM.

[3] Brody, S., & Elhadad, N. (2010). An unsupervised aspect-sentiment model for online reviews. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics* (pp. 804-812). Association for Computational Linguistics.

[4] Engonopoulos, N., Lazaridou, A., Paliouras, G., & Chandrinos, K. (2011). ELS: a word-level method for entity-level sentiment analysis. In *Proceedings of the International Conference on Web Intelligence, Mining and Semantics*

[5] Moilanen, K., & Pulman, S. (2009). Multi-entity Sentiment Scoring. In *Recent Advances in NLP* (pp. 258-263).

[6] Lakkaraju, H., Socher, R, & Manning, C. (2014). Aspect Specific Sentiment Analysis using Hierarchical Deep Learning. *NIPS Workshop on Deep Learning and Representation Learning*

[7] Socher, R., Perelygin, A., Wu, J. Y., Chuang, J., Manning, C. D., Ng, A. Y., & Potts, C. (2013). Recursive deep models for semantic compositionality over a sentiment treebank. In *Proceedings of the conference on Empirical Methods in Natural Language Processing (EMNLP)* (Vol. 1631, p. 1642).

[8] McAuley, J., Targett, C., Shi, J., & van den Hengel, A. (2015). Image-based recommendations on styles and substitutes. *ACM Special Interest Group on Information Retrieval (SIGIR)*

[9] Bird, Steven, Loper, E. and Klein, E. (2009). Natural Language Processing with Python. OReilly Media Inc.