UGSD: User Generated Sentiment Dictionaries from Online Customer Reviews

Chun-Hsiang Wang,[1] **Kang-Chun Fan,**[2] **Chuan-Ju Wang,**[2] **Ming-Feng Tsai**[1,3]
[1]Department of Computer Science, National Chengchi University, Taiwan
[2]Research Center for Information Technology Innovation, Academia Sinica, Taiwan
[3]MOST Joint Research Center for AI Technology and All Vista Healthcare, Taiwan
{ch_wang, mftsai}@nccu.edu.tw, {kcfan, cjwang}@citi.sinica.edu.tw

Abstract

Customer reviews on platforms such as TripAdvisor and Amazon provide rich information about the ways that people convey sentiment on certain domains. Given these kinds of user reviews, this paper proposes UGSD, a representation learning framework for constructing domain-specific sentiment dictionaries from online customer reviews, in which we leverage the relationship between user-generated reviews and the ratings of the reviews to associate the reviewer sentiment with certain entities. The proposed framework has the following three main advantages. First, no additional annotations of words or external dictionaries are needed for the proposed framework; the only resources needed are the review texts and entity ratings. Second, the framework is applicable across a variety of user-generated content from different domains to construct domain-specific sentiment dictionaries. Finally, each word in the constructed dictionary is associated with a low-dimensional dense representation and a degree of relatedness to a certain rating, which enable us to obtain more fine-grained dictionaries and enhance the application scalability of the constructed dictionaries as the word representations can be adopted for various tasks or applications, such as entity ranking and dictionary expansion. The experimental results on three real-world datasets show that the framework is effective in constructing high-quality domain-specific sentiment dictionaries from customer reviews.

Introduction

As more and more people share their experience and thoughts on platforms including forums, review websites, and microblogs, there has been a surge of research on sentiment analysis. Sentiment analysis, the goal of which is to identify the sentiment of a text, has been applied to a variety of data sources. For example, one study uses Twitter as their corpus to build a sentiment classifier to determine positive, negative, and neutral sentiments (Pak and Paroubek 2010), while another line of research mines opinions about entities in the news, which are different from subjective text types (Balahur et al. 2010). Due to these studies, sentiment analysis can be adopted in various applications, including real-world event monitoring and public opinion mining. In addition, with the growing popularity of online reviews and e-commerce websites such as TripAdvisor, Yelp, and Amazon, sentiment analysis on the reviews posted on these platforms is crucial since it has been confirmed that they have a direct influence on commercial revenue and reputation for businesses (Luca 2011; Luca and Zervas 2016; Guzman and Maalej 2014).

For sentiment analysis, the dictionary is one of the most vital resources and generally has a great influence on results and the corresponding analyses (Feldman 2013). However, it is difficult to construct a universal sentiment dictionary that fits all domains because word sentiment in general depends on the domain it is being used for. In particular, dictionaries of one domain cannot easily be applied to other domains, because words can have opposite sentiments when used in different situations (Lu et al. 2011; Balahur et al. 2010; Hamilton et al. 2016). For instance, the word "soft" has a positive meaning for burgers, but conveys a negative attitude toward basketball players. Consequently, one important task in the fields of natural language processing and text mining is constructing domain-specific sentiment dictionaries.

There have been many studies on automatic methods for building the dictionaries for various domains (Hamilton et al. 2016; Labille, Gauch, and Alfarhood 2017). Most proposed approaches either manually or semi-automatically leverage other resources and calculate the scores of new words via the proximity to one or more seed words, i.e., a small set of words with strong positive or negative associations (Taboada et al. 2011). For example, (Hamilton et al. 2016) induce domain-specific sentiment dictionaries by building high-quality semantic representations and propagating the polarity score of each word from a seed set with random walks. Another study utilizes online reviews to construct context-aware dictionaries, applying a generic sentiment dictionary and synonym-antonym dictionary, and exploiting linguistic heuristics for overall review ratings (Lu et al. 2011). In sum, most previous studies require general sentiment dictionaries, annotated seed words, or linguistic heuristics; clearly, these external resources greatly impact the results of dictionary construction. One recent study does not use a generic dictionary (Labille, Gauch, and Alfarhood 2017), which however considers binary sentiments only and ignores all reviews with medium ratings.

The explosion of user-generated data on the web, including customer reviews on platforms like TripAdvisor and e-commerce websites like Amazon, constitutes a great oppor-

tunity to attain rich information about the ways that people convey their sentiment in certain domains. Therefore, we propose UGSD, a representation learning framework for constructing sentiment dictionaries from customer reviews. The proposed framework requires no external resources such as annotated seed words or general-purpose dictionaries and leverages the relationship between the texts of user-generated reviews and the corresponding ratings to associate the sentiment of reviewers with certain entities. Specifically, there are four main parts in our framework: 1) candidate sentiment word selection, which leverages part-of-speech (POS) information and named-entity recognition techniques; 2) review transformation, which replaces entities in reviews with corresponding rating symbols; 3) word representation learning, which models word co-occurrence proximity; 4) finally, lexicon construction, in which a sentiment dictionary is formed for each rating via the learned representations.

To evaluate the effectiveness of the proposed framework, we conduct extensive experiments on three real-world datasets: Yelp restaurant reviews, TripAdvisor attraction reviews, and Amazon product reviews. Three types of experiments are conducted: 1) We compare the generated Yelp dictionaries with the state-of-the-art Stanford Yelp dictionary (Reschke, Vogel, and Jurafsky 2013); we conduct both 2) the traditional sentiment classification and 3) the entity ranking (Chao et al. 2017) to evaluate the effectiveness of the generated dictionaries. The experimental results show that the framework is effective in constructing high-quality, domain-specific sentiment dictionaries from online reviews. In summary, our framework advances the state of the art in the following three dimensions.

1. Data-driven dictionary: Requiring no additional annotation of seed words or external dictionaries.
2. Domain-specific dictionaries: Applying to a variety of user-generated content from different domains to construct domain-specific sentiment dictionaries.
3. Application scalability: Producing representations of the learned sentiment words during the dictionary construction, which enable us to obtain more fine-grained dictionaries and enhance the usability of the constructed dictionaries for various applications.

Methodology

Definitions and Problem Formulation

We first define the notations used in the paper (see Table 1) and formulate the dictionary generation problem. We then provide an overview of the proposed framework and offer a remark at the end of this section.

With the definitions in Table 1, we here provide a formal description of our dictionary generation problem.

Definition 1. (Dictionary Generation Problem) *Given a set of reviews of a certain domain $\mathcal{D}$ and the set of entities $\mathcal{E}$ evaluated in all reviews in $\mathcal{D}$, each pair (d, e), where $d \in \mathcal{D}$ and $e \in \mathcal{E}$, corresponds to a rating symbol $r \in \mathcal{R}$ by a mapping function $\mathcal{M}$. With $\mathcal{D}$, $\mathcal{E}$, and $\mathcal{M}$, the sentiment dictionary generation problem aims to, for each of the rating symbols $r \in \mathcal{R}$, generate a sentiment dictionary $\mathcal{L}_r$ comprising a set of tuples, each of which $(s, \vec{v}_s, \theta_s^r)$ contains a sentiment word s, its representation $\vec{v}_s$, and its degree of relatedness θ_s^r to the rating symbol r.*

Table 1: Notation definitions

Notation	Meaning
$\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$	A set of reviews in a certain domain
$d_i = w_i^1 w_i^2 \ldots w_i^{M_i}$	A sequence of words $w_i^1 \ldots w_i^{M_i}$ of review d_i; the length of d_i is M_i
$\mathcal{E} = \{e_1, e_2, \ldots, e_K\}$	A set of entities mentioned in all of the reviews in $\mathcal{D}$
$\mathcal{S} = \{s_1, s_2, \ldots, s_G\}$	A set of candidate sentiment words
$\mathcal{R} = \{r_1, r_2, \ldots, r_H\}$	A set of rating symbols, each of which corresponds to a unique rating
$\mathcal{M}(d, e)$	A mapping function $\mathcal{M} : \mathcal{D} \times \mathcal{E} \rightarrow \mathcal{R}$ that maps a paired review $d \in \mathcal{D}$ and entity $e \in \mathcal{E}$ evaluated in the review to a rating symbol $r \in \mathcal{R}$
$(s, \vec{v}_s, \theta_s^r)$	A tuple denoting a sentiment word s, its representation $\vec{v}_s$, and its degree of relatedness θ_s^r to the rating symbol r
$\mathcal{L}_r$	A set of tuples $(s, \vec{v}_s, \theta_s^r)$, each of which corresponds to a sentiment word s selected from the candidate set $\mathcal{S}$ for the rating symbol r

Take for example the TripAdvisor reviews for attractions. In this scenario, an entity is an attraction, and a review $d \in \mathcal{D}$ is written by a user for only one specific attraction $e \in \mathcal{E}$; therefore, the function $\mathcal{M}(d, e)$ simply gives the rating associated with the review d. Given thousands of reviews with their ratings for various attractions, for each of the rating symbols $r \in \mathcal{R}$, the proposed framework produces a set of sentiment words with their representations and degrees of relatedness to r in the form of $(s, \vec{v}_s, \theta_s^r)$ by exploiting both the text and rating information in the reviews.

In other words, our framework attempts to exploit the reviews and their corresponding ratings from users on online review websites such as TripAdvisor, Amazon, and Yelp to automatically generate domain-specific sentiment dictionaries. An overview of the proposed framework is provided as follows. First, we select a set of candidate sentiment words $\mathcal{S}$ by leveraging POS information and in the meantime address the ambiguity of named entities and the commonly appearing negation problem. We then introduce a sequence transformation method to replace entities in reviews with the corresponding rating symbols. After that, we define the co-occurrence proximity between words and leverage this proximity to learn the representations of words and rating symbols. Finally, we explain how we construct the dictionary $\mathcal{L}_r$ for each unique rating $r \in \mathcal{R}$.

Remark Note that a review can mention more than one entity, each of which is sometimes associated with its own rating; for example, users may provide different ratings for different aspects of a hotel or for different dishes of a restaurant. The quality of the generated dictionaries, especially those associated with intermediate ratings, thus depends on the granularity of ratings provided in a customer review

since the intuition of the proposed framework is that the preference of an entity is coupled with its co-occurrence with sentiment words used to describe it in the textual proximity. Due to the characteristics of the datasets used in the experiments, we only consider cases for which a review is associated with a single rating symbol; it however poses no problem to apply the proposed framework to a more fine-grained dataset, if available.

Representation Learning Framework

Candidate Sentiment Word Selection We show how to select a set of candidate sentiment words $\mathcal{S}$ by leveraging POS information; in the meantime we address the ambiguity of named entities and the commonly appearing negation problem. Following most previous work, we select adjectives and adverbs from all reviews for use as the candidate words for sentiment dictionary construction (Lu et al. 2011; Pak and Paroubek 2010). Since adverbs sometimes intensify or diminish adjectives (Ruppenhofer et al. 2015; Taboada et al. 2011), if adverbs are followed by adjectives, our framework adopts commonly used linguistic heuristics to combine them using the *adverb_adjective* form. This approach links the meaning of the adverbs to the adjective degree and thus scales the sentiments of different adverb-adjective combinations. Moreover, words used as function words to make sequential words negative – such as "not" and "never" – totally reverse reviewer sentiment. In this way to concatenate adverbs and adjectives, the negation problem is generally solved at the same time as such negative words are usually part-of-speech tagged as adverbs. The combinations are thus the *not_adjective* or *never_adjective* forms, also frequently adopted in the review analysis literature (Reschke, Vogel, and Jurafsky 2013).

On the other hand, except for targeted entities in $\mathcal{E}$, there are abundant other named entities, such as the names of organizations, people, or locations, in reviews. These named entities are part-of-speech tagged as adjectives if they are used as possessives and thus are included in the candidate sentiment words, which however do not imply user sentiments. Thus, our framework solves this problem by leveraging named-entity recognition techniques to exclude them from candidate sentiment words. Finally, we utilize a set of candidate sentiment words $\mathcal{S}$ to serve as a source for the construction of the sentiment dictionary in later phases, detailed settings for which are provided in the experiment section.

Review Transformation Ratings from users tend to imply emotions toward the entities evaluated. In other words, a high rating associated with an entity in a customer review generally means that the customer has a positive attitude toward that entity and tends to describe it with positive sentiment words. We thus leverage the sentiment words surrounding the described entities in reviews in the proposed dictionary generation framework.

To achieve this goal, for each of the reviews $d_i \in \mathcal{D}$, our framework transforms the review by replacing each of the mentioned entities $e \in \mathcal{E}$ with the corresponding rating symbol $r \in \mathcal{R}$ via the given mapping function $\mathcal{M}(d_i, e)$. Formally speaking, it can be generalized as a sequence transformation, i.e., an operator acting on a given space of sequences. For a given word sequence $d_i = w_i^1 w_i^2 \dots w_i^{M_i}$, the transformed sequence is

$$d_i' := \mathcal{T}(d_i) = {w'}_i^1 {w'}_i^2 \dots {w'}_i^{M_i}, \tag{1}$$

where the members of the transformed sequence ${w'}_i^k$ are computed from the corresponding members of the original sequence as

$${w'}_i^k = \begin{cases} \mathcal{M}(d_i, e) & \text{if } w_i^k = e \in \mathcal{E} \\ w_i^k & \text{otherwise.} \end{cases} \tag{2}$$

After the above transformation, each entity is expressed with a rating symbol $r \in \mathcal{R} = \{r_1, r_2, \dots, r_H\}$. Note that the range of ratings varies in accordance with different datasets, resulting in different cardinalities for set $\mathcal{R}$. With the variance of ratings associated with entities in reviews and the co-occurrence proximity between rating symbols and words (described later), our framework enables us to construct more fine-grained and data-oriented dictionaries toward positive and negative opinions expressed in different contexts.

Co-occurrence Proximity Learning

Definition 2. (k Co-occurrence Proximity) The co-occurrence proximity refers to the pairwise proximity between the words in a corpus $\mathcal{D}$ comprising a set of unique words $\mathcal{V}$. The co-occurrence frequency $f_{ij} \geq 0$ between two unique words, $i, j \in \mathcal{V}$, denotes the frequency of word j occurring in the context of word i within a predefined window size k calculated from all documents in $\mathcal{D}$; this is used to quantify the k co-occurrence proximity for the word pair $(i, j) \in \mathcal{A}$, where $\mathcal{A}$ denotes the set of all word pairs.

Given a set of reviews transformed via Equations (1) and (2), $\mathcal{D}' = \{d_1', d_2', \dots, d_N'\}$, we define the k co-occurrence proximity between all pairs of words in the set of unique vocabularies $\mathcal{V}$ in $\mathcal{D}'$. Each of the words in $\mathcal{V}$ belongs to one of the following three disjoint sets: 1) the set of rating symbols $\mathcal{R}$, 2) the set of candidate sentiment words $\mathcal{S}$, and 3) the set of all other words $\mathcal{V} - \mathcal{R} - \mathcal{S}$. The goal of representation learning is to presume this co-occurrence proximity between words in the given review corpus and obtain their embedding vectors; that is, words with strong co-occurrence proximities are correlated and thus, when represented in a low-dimensional vector space, should be positioned close to one another. For instance, if a sentiment word is always used to describe an entity, which means the sentiment word frequently surrounds the entity, and their representations are thus closely located in the learned vector space.

In this paper, we model the co-occurrence proximity of words by learning the low-dimensional representations for words in $\mathcal{V}$. With the defined k co-occurrence proximity, for each word pair (i, j), the joint probability between words i and j is defined as (Tang et al. 2015)

$$p(i, j) = \frac{1}{1 + e^{-\vec{v}_i^{\intercal} \cdot \vec{v}_j}}, \tag{3}$$

where the low-dimensional vector $\vec{v}_c \in \mathbb{R}^d$ denotes the representation of word c with $d \ll |\mathcal{V}|$. Above, Equation (3) defines the joint distribution with the function $p(\cdot, \cdot)$, the

empirical probability of which is denoted as $\hat{p}(i,j)$ and can be set to $f_{ij}/\mathcal{F}$, where $\mathcal{F} = \sum_{(i,j)\in\mathcal{A}} f_{ij}$. Recall that f_{ij} denotes the co-occurrence frequency between two unique words, $i, j \in \mathcal{V}$ and $\mathcal{A}$ denotes the set of all word pairs (i,j).

One way to preserve the co-occurrence proximity is to minimize the distance between the empirical and learned distributions, i.e., $\hat{p}(\cdot,\cdot)$ and $p(\cdot,\cdot)$; we construct the objective function as $O = dist(\hat{p}(\cdot,\cdot), p(\cdot,\cdot))$, where $dist(\cdot,\cdot)$ denotes the distance between two distributions. Replacing the distance function with the commonly used Kullback-Leibler divergence as in (Tang et al. 2015; Tang, Qu, and Mei 2015; Wang et al. 2017; Chao et al. 2017), we have

$$O = -\sum_{(i,j)\in\mathcal{A}} f_{ij} \log p(i,j). \tag{4}$$

The objective function (4) is minimized using stochastic gradient descent with edge sampling (Tang et al. 2015) and negative sampling (Mikolov et al. 2013), which results in a set of optimized representations $\{\vec{v}_c\}$ for all words $c \in \mathcal{V}$. The resulting low-dimensional vectors are vital, retaining the relationships between rating symbols and other words; these relationships are useful, as they can be further utilized in the next section for the calculation of distances between each rating symbol and all other words, and in turn the construction of the dictionaries.

Dictionary Construction With the learned vector representations of words and rating symbols, in the last phase of the proposed framework we first define a matrix A documenting the cosine similarity between each candidate word $s_i \in \mathcal{S}$ and each rating symbol $r_j \in \mathcal{R}$ as

$$A = \begin{pmatrix} a_{11} & a_{12} & \cdots & a_{1|\mathcal{R}|} \\ a_{21} & a_{22} & \cdots & a_{2|\mathcal{R}|} \\ \vdots & \vdots & \ddots & \vdots \\ a_{|\mathcal{S}|1} & a_{|\mathcal{S}|2} & \cdots & a_{|\mathcal{S}||\mathcal{R}|} \end{pmatrix}, \tag{5}$$

where $a_{ij} = \cos(\vec{v}_{s_i}, \vec{v}_{r_j})$.

With the similarity matrix A in Equation (5), we can design an element-wise function $\mathcal{G}(A)$ that maps each candidate sentiment word to rating symbols as

$$\mathcal{G}(A) = (b_{ij}) \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{R}|}, \text{ where } b_{ij} \in \{0,1\}, \tag{6}$$

where $b_{ij} = 1$ indicates that the candidate sentiment word s_i belongs to the class of the rating symbol r_j, otherwise $b_{ij} = 0$. With $\mathcal{G}(A)$ in Equation (6), the sentiment dictionary $\mathcal{L}_{r_j}$ with respect to the rating symbol r_j becomes

$$\mathcal{L}_{r_j} = \left\{ (s_i, \vec{v}_{s_i}, \theta^{r_j}_{s_i} = a_{ij}) \,|\, b_{ij} = 1 \right\}.$$

The $\mathcal{G}(\cdot)$ can be designed in various ways. In this paper, we provide two heuristics with simple statistics as follows.

A maximum-cosine-similarity scheme In this selection scheme, for each $s_i \in \mathcal{S}$, we define the maximum cosine value with respect to each rating symbol $r_j \in \mathcal{R}$ (i.e., the maximum value in each row of A in Equation (5)) as

$$m_i = \max_{1\le j\le|\mathcal{R}|} a_{ij}.$$

Then, we have

$$\mathcal{G}_{\max}(A) = (b_{ij}) = \left(\mathbb{1}_{\{a_{ij}\ge m_i\}}\right) \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{R}|}.$$

The intuition behind this selection scheme is that we assign a word to the rating class to which the word is with the highest degree of relatedness; this results in a one-to-one mapping scenario for each candidate sentiment word.

A z-score scheme In this selection scheme, we use the z-scores calculated from the cosine similarities of candidate sentiment words with respect to each rating symbol $r_j \in \mathcal{R}$ to determine the dictionary $\mathcal{L}_{r_j}$. For each a_{ij} in matrix (5), the z-score, representing the distance between the cosine similarity a_{ij} and the column mean μ_j in units of the column standard deviation σ_j, is $z_{ij} = (a_{ij} - \mu_j)/\sigma_j$, where $\mu_j = \sum_{i=1}^{|\mathcal{S}|} a_{ij}/|\mathcal{S}|$ and $\sigma_j = \sqrt{(\sum_{i=1}^{|\mathcal{S}|}(a_{ij} - \mu_j)^2)/|\mathcal{S}|}$. Then, we have

$$\mathcal{G}_{z>\ell}(A) = (b_{ij}) = \left(\mathbb{1}_{\{z_{ij}>\ell\}}\right) \in \mathbb{R}^{|\mathcal{S}|\times|\mathcal{R}|},$$

where ℓ is a predefined threshold. The reason why we choose column-wise z-scores instead of row-wise z-scores to determine the dictionaries is that for each word (i.e., each row), it is only a few cosine similarities to calculate the statistics, which may result in non-representative or non-stable values. Note that a word may belong to more than one classes of rating symbols in this scenario.

The learned representations and the cosine similarity matrix (5) allow us to generate a set of representative sentiment words for each rating symbol and thus produce opinion dictionaries of different granularity. Moreover, different selection schemes may result in dictionaries with different characteristics that are experimented and discussed later.

Experiments

Data Description

Yelp Restaurant Reviews The customer reviews of the Yelp dataset were collected from the 9th round of the Yelp Dataset Challenge,[1] from which we extracted the reviews of 215 restaurants located in Las Vegas, as the Vegas area has the most reviews as compared to other areas in the challenge dataset. Although this dataset contains user-generated reviews and ratings, the main entities mentioned in the review of a restaurant, the dishes provided in each restaurant, are not provided by the challenge; we thus manually scraped the menus from the Yelp official website and treated the dishes as entities to be later replaced with rating symbols. In the Yelp dataset, the 5-level ratings range from 1 to 5 stars (i.e., $\mathcal{R} = \{r_1, r_2, \ldots, r_5\}$). Note that as the rating for a Yelp review is associated with a restaurant and not with a dish, we here map the dishes described by a user in a same review to the rating associated with the review.

TripAdvisor Attraction Reviews The dataset was collected from TripAdvisor's official website, where the top 25 cities in 2016 were selected, and the reviews of the top 20 attractions or tours of each city were included. Each attraction

[1] https://www.yelp.com/dataset/challenge

Table 2: Statistics for the three real-world datasets

	Yelp	TripAdvisor	Amazon
# review	186,834	2,870,024	123,526
Avg. # reviews per entity	869	5,740	1,029
Avg. review length (words)	121	81	108
# vocabularies (before preprocessing)	396,587	2,086,442	330,035
Avg. senti-entity proximal distance	1.668	1.544	1.532
# candidate sentiment words	3,607	4,196	2,262

or tour in a city comprises its rating statistics and user reviews, each of which is composed of the user-generated text and a user rating ranging from 1 to 5 stars. Note that all of the tours consist of numerous sub-tours on the TripAdvisor official website; thus we gathered and merged the reviews of the sub-tours into a single tour. In this dataset, each attraction or tour was treated as our entity.

Amazon Product Reviews This dataset provided by (Wang, Lu, and Zhai 2010) consists of reviews on six categories of electronic supplies: cameras, televisions, laptops, mobile phones, tablets, and video surveillance equipment. Each category contains various products that were treated as our entities. To conduct the experiments, we extracted the reviews for the top 20 products with the most reviews for each category. Similar to TripAdvisor, each product in a category contains its rating statistics and customer reviews, each of which includes user-generated text and a user rating ranging from 1 to 5 stars.

Data Preprocessing and Experiment Setup

Data Preprocessing Given a list of entities (attractions, product names, or dishes), it is necessary to first recognize these entities in online reviews; this however can be difficult due to noise from inconsistent language usage on the Internet, as users tend to abbreviate the names of entities or in many cases use only the last word or the last two words of an entity name in reviews. To remove the ambiguity of different expressions from separate users, we created our own parser that automatically generates regular expressions based on the names of entities and several observed features of reviews to identify these entities. In addition, per our findings, users tend to mention the names of entities or simpler names once or twice, and then use pronouns or other expressions to note the same entities in reviews. However, as it is difficult to cover these using universal regular expressions, we further perform coreference resolution with the dcoref annotator from Stanford CoreNLP (Manning et al. 2014) to extract other expressions which also refer to the targeted entities and thus improve the quality of entity recognition.[2]

Candidate Sentiment Word Retrieval We first tag POSs and remove named entities with CoreNLP (Manning et al. 2014); then, from all reviews we sift out those adjectives and adverbs that occur more than 100 times for TripAdvisor and 20 times for Amazon and Yelp, respectively. Specifically, to extract the initial batch of sentiment words, our framework draws out all adjectives and adverbs with POS tags `JJ`, `JJR`, `JJS`, `RB`, `RBR`, and `RBS`.[3] After extracting candidate words from the corpus, the stop words[4] are removed, after which our framework processes the reviews to obtain three types of candidate sentiment words in the form of *adverb*, *adjective*, and *adverb_adjective*. Then, the remaining words are then stemmed using the Snowball stemmer. The resulting numbers of candidate sentiment words for the three datasets are reported in Table 2.

Model Parameter Selection Whether a co-occurrence proximity is well-modeled depends on the proper estimation of the influence of the proximal words; thus, we compute the average distance of each entity and its nearest candidate sentiment word, which results in distances of 1.668, 1.544, and 1.532 in the Yelp, TripAdvisor, and the Amazon datasets, respectively (see Table 2). We hence select the window size $k = 2$ for all three datasets in the experiments. Particularly, we assemble the reviews from each dataset into a single text file to measure the co-occurrence proximity, truncating co-occurrence pairs where either word occurs fewer than fives times in the corpus. To learn embeddings to presume the co-occurrence proximity, the number of negative samples is set to 5, the representation dimension is set to 200, and the total number of samples is set to 25 million.

Quantitative Evaluation

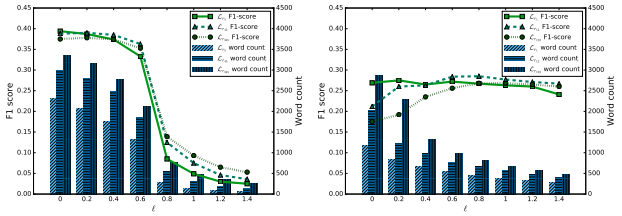
Comparison with Standford Yelp Dictionary For Yelp dataset, we used the proposed method to generate the sentiment dictionaries, and then compared the dictionaries with three general-purpose dictionaries (NLTK Opinion dictionary (Hu and Liu 2004), MPQA Subjectivity dictionary (Wiebe, Wilson, and Cardie 2005), and SentiWordNet (Baccianella, Esuli, and Sebastiani 2010)) to that constructed by the Stanford NLP group using graph propagation (Reschke, Vogel, and Jurafsky 2013), which is considered as the state-of-the-art sentiment dictionaries for Yelp. The dataset collection used in (Reschke, Vogel, and Jurafsky 2013) contains 11,537 businesses with 229,907 reviews, and the resulting Yelp-specific sentiment dictionary (denoted as Stanford Yelp dictionary hereafter) contains 1,435 positive and 570 negative words. To simplify the notation, henceforth, $\mathcal{L}_{r_{ij}}$ (or $\mathcal{L}_{r_{ijk}}$) denotes the union of the two dictionaries $\mathcal{L}_{r_i}$ and $\mathcal{L}_{r_j}$ (or the union of the three dictionaries $\mathcal{L}_{r_i}$, $\mathcal{L}_{r_j}$, and $\mathcal{L}_{r_k}$, respectively). To compare with the Stanford Yelp dictionaries, we first conducted a sensitivity analysis on the z-score scheme with different thresholds ℓ for constructing the dictionaries. Figure 1 plots the results, where the curves denote the F1-score of comparing the generated dictionaries to the Stanford Yelp dictionary, and the bars indicate the word count of the dictionaries. For the positive words of the Stanford Yelp dictionary, we used the dictionaries $\mathcal{L}_{r_5}$, $\mathcal{L}_{r_{45}}$, and $\mathcal{L}_{r_{345}}$ for comparison; on the other hand, the dictionaries $\mathcal{L}_{r_1}$, $\mathcal{L}_{r_{12}}$, and $\mathcal{L}_{r_{123}}$ were used for negative

[2]Using only regular expressions yields an average 60.09% F1 score for entity recognition; incorporating coreference resolution achieves an average 85.91% F1 score.

[3]We use the Penn Treebank tag set.

[4]We use the stop word list provided by NLTK.

Figure 1: Sensitivity analysis with respect to Stanford Yelp sentiment dictionaries



(a) Positive dictionary (b) Negative dictionary

Table 3: Precision, recall, and F1-score with respect to Stanford Yelp sentiment dictionaries

		Positive				Negative			
		# word	P	R	F1	# word	P	R	F1
	NLTK	2,006	0.196	0.275	0.229	4,783	0.072	**0.607**	0.129
	MPQA	2,304	0.198	0.318	0.244	4,152	0.079	0.579	0.139
	SentiWordNet	14,712	0.039	0.395	0.071	10,751	0.015	0.288	0.029
$\mathcal{G}_{\max}(\cdot)$	$\mathcal{L}_{r_5}$	594	**0.352**	0.146	0.206	$\mathcal{L}_{r_1}$ 1,112	0.161	0.314	0.213
	$\mathcal{L}_{r_{45}}$	1,125	0.332	0.260	0.292	$\mathcal{L}_{r_{12}}$ 1,901	0.140	0.467	0.215
	$\mathcal{L}_{r_{345}}$	1,685	0.315	0.369	0.340	$\mathcal{L}_{r_{123}}$ 2,461	0.119	0.512	0.193
$\mathcal{G}_{z>0.6}(\cdot)$	$\mathcal{L}_{r_5}$	1,309	0.349	0.318	0.333	$\mathcal{L}_{r_1}$ 534	**0.281**	0.263	0.272
	$\mathcal{L}_{r_{45}}$	1,860	0.322	0.417	**0.363**	$\mathcal{L}_{r_{12}}$ 773	0.247	0.335	**0.284**
	$\mathcal{L}_{r_{345}}$	2,113	0.296	**0.436**	0.353	$\mathcal{L}_{r_{123}}$ 990	0.202	0.351	0.256

word comparison. As shown in Figure 1(a), there is a drop of the F1-scores from $\ell = 0.6$ to $\ell = 0.8$;[5] therefore, we choose $\ell = 0.6$ as the threshold setting for the following comparisons with other baseline dictionaries.

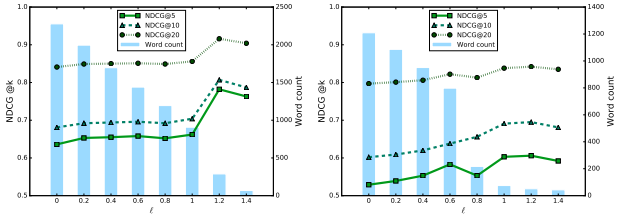
Table 3 compares our dictionaries built by the maximum-cosine-similarity scheme and the z-score scheme with $\ell = 0.6$ and the three general-purpose dictionaries with the positive and negative words in the Stanford Yelp dictionary. As shown in the table, all of the combinations of our dictionaries built by the maximum-cosine-similarity and z-score schemes both outperform all the baselines in terms of precision; additionally, there are 11 out of the total 12 combinations of our dictionaries getting better F1-scores than the three baseline dictionaries. Moreover, observe that the recall of the dictionary $\mathcal{L}_{r_{123}}$ with only 2,461 words built by the maximum-cosine-similarity scheme achieves 0.512, which is close to the recall 0.607 of the negative opinion dictionary, which has about twice word count (i.e., 4,783 words). Similarly, $\mathcal{L}_{r_{345}}$ with only 2,113 words constructed by the z-score scheme beats the best recall of the baselines, SentiWordNet with 14,712 positive words. These phenomena again manifest that our proposed framework can generate effective domain-specific sentiment dictionaries. It is also worth mentioning that our framework selects adverbs and adjectives as the candidate sentiment words, whereas the Stanford Yelp dictionary includes adjectives only.

[5]Note that in this case, the word count drops from 1,309 to 279.

Table 4: Performance on sentiment classification

		Yelp			TripAdvisor			Amazon		
		# word	F1	Acc	# word	F1	Acc	# word	F1	Acc
	NLTK	6,787	0.762	0.697	6,787	**0.759**	0.699	6,787	0.766	0.707
	MPQA	6,450	0.708	0.601	6,450	0.701	0.608	6,450	0.716	0.616
	SentiWordNet	24,123	0.675	0.534	24,123	0.670	0.520	24,123	0.685	0.551
	Stanford Yelp	2,005	0.682	0.534	2,005	0.686	0.544	2,005	0.679	0.530
$\mathcal{G}_{\max}(\cdot)$	$\mathcal{L}_{r_5} \cup \mathcal{L}_{r_1}$	1,524	0.733	0.755	1,888	0.664	0.679	717	0.744	0.727
	$\mathcal{L}_{r_{45}} \cup \mathcal{L}_{r_{12}}$	2,692	0.771	**0.777**	3,428	0.746	**0.753**	1,566	0.763	0.755
$\mathcal{G}_{z>1.2}(\cdot)$	$\mathcal{L}_{r_5} \cup \mathcal{L}_{r_1}$	364	0.784	0.758	710	0.726	0.630	189	**0.801**	**0.782**
	$\mathcal{L}_{r_{45}} \cup \mathcal{L}_{r_{12}}$	451	**0.792**	0.762	1,060	0.736	0.650	346	0.800	0.772

Figure 2: Sensitivity analysis on entity ranking



(a) TripAdvisor (b) Amazon

Sentiment Classification We conducted the binary sentiment classification by a straightforward approach, where the term frequencies of positive and negative words were calculated to identify the sentiment of reviews (Labille, Gauch, and Alfarhood 2017; Lu et al. 2011; Hamilton et al. 2016). In this experiment, we take reviews that are not used to produce our dictionaries to evaluate the performance; specifically, for Yelp and Amazon, the testing datasets are from the remaining reviews of the Yelp Dataset Challenge and (Wang, Lu, and Zhai 2010), and for TripAdvisor, we collected the reviews of other four cities: San Francisco, Los Angeles, Seattle, and Boston from TripAdvisor's official website. We filtered out the reviews with the 3 star; thus, the 4 and 5 star reviews were considered as positive reviews and the 1 and 2 star reviews were considered as negative ones. Table 4 displays the performance on three datasets. For Yelp, our dictionaries built by both schemes outperform all baselines with respect to the F1-score and accuracy; specifically, all our dictionaries listed in the table achieve much better performance than the domain-specific Stanford Yelp dictionary. For TripAdvisor, our dictionary produced by the maximum-cosine-similarity scheme obtains the best accuracy and the comparable F1-score to the best performed NLTK, with much fewer words in our dictionaries. At last, for Amazon, our dictionaries again achieve superior performance than all of the other baseline dictionaries. These results attest that our framework has the ability to produce high-quality domain-specific dictionaries.

Table 5: Performance on entity ranking

		TripAdvisor			Amazon		
		# word	NDCG@5	NDCG@10	# word	NGCG@5	NDCG@10
	Frequency	-	0.610	0.664	-	0.494	0.623
	NLTK	1,071	0.556	0.632	595	0.603	0.659
	MPQA	1,294	0.562	0.641	710	0.571	0.654
	SentiWordNet	4,522	0.442	0.530	2,207	0.543	0.574
$\mathcal{G}_{\max}(\cdot)$	$\mathcal{L}_{r_5}$	207	**0.794**	**0.818**	258	**0.635**	**0.712**
	$\mathcal{L}_{r_{45}}$	745	0.669	0.724	493	0.549	0.641
	$\mathcal{L}_{r_{345}}$	1,626	0.654	0.698	995	0.574	0.655
$\mathcal{G}_{z>1.2}(\cdot)$	$\mathcal{L}_{r_5}$	288	0.782	0.807	51	0.606	0.695
	$\mathcal{L}_{r_{45}}$	569	0.735	0.770	114	0.515	0.631
	$\mathcal{L}_{r_{345}}$	895	0.719	0.751	221	0.515	0.627

Entity Ranking We conducted an entity ranking task on TripAdvisor and Amazon[6] and the ranking method involves only positive sentiment words (Chao et al. 2017). So, we compared our three dictionaries, $\mathcal{L}_{r_5}$, $\mathcal{L}_{r_{45}}$, $\mathcal{L}_{r_{345}}$, with positive words in the three generic dictionaries. For each entity we regarded the average rating stars from the customer review as the ground truth; additionally, the performance of the ranking task was measured in terms of normalized discounted cumulative gain (NDCG), for which we labeled the entities from 4 to 1 according to their ground truth stars.

We first analyzed the sensitivity to the threshold values ℓ for the z-score scheme in $\mathcal{L}_{r_5}$, as in Figure 2 where the curves show the NDCG scores and the bars exhibit the word counts of $\mathcal{L}_{r_5}$.[7] We found using more words to construct the positive sentiment dictionary does not always translate to better ranking performance in terms of NDCG, and the two $\mathcal{L}_{r_5}$ dictionaries with the $\ell = 1.2$ both achieve the best performance. The sensitivity analysis also exemplifies the advantage of the proposed framework that our framework generated ranked lists of words with different degrees of strength to form a sentiment dictionary.

Table 5 tabulates the ranking performance in terms of NDCG@5 and @10 for TripAdvisor and that for Amazon, which is the result of the dictionaries built by the maximum-cosine-similarity scheme and the z-score scheme with the threshold $\ell = 1.2$. We compared the results with four baseline methods: the first one, frequency, ranks entities via their occurrence counts in the total reviews of the corresponding city or product category; the others leverage the ranking function defined in (Chao et al. 2017) with different generic dictionaries.[8] Our dictionaries $\mathcal{L}_{r_5}$, produced by the maximum-cosine-similarity scheme, yields superior average performance over the four baselines by a significant amount for both datasets. On the other hand, for the z-score selection scheme, the dictionary $\mathcal{L}_{r_5}$ also performs better than all the baselines in average. Although the dictionaries $\mathcal{L}_{r_{45}}$ and $\mathcal{L}_{r_{345}}$ do not perform as well as $\mathcal{L}_{r_5}$, they still obtain better average performance than the four baselines. This phenomenon indicates that the dictionaries associated with lower ratings may deteriorate the ranking performance as some of the less representative positive sentiment words are included and our framework effectively generates sentiment dictionaries with different granularity. Finally, as a side note, for the entity ranking, we used only a small number of words in each dictionary to rank entities and outperform all baselines, whereas the generic dictionaries contain much more words, showing the quality of the dictionary is much more important than its size and also attest the high quality of the domain specific dictionaries generated by our framework.

[6]We did not conduct the entity ranking for Yelp due to the unavailability of the ground truth for its entities (i.e., dishes).

[7]Due to the page limit, we here list the analysis only for $\mathcal{L}_{r_5}$.

[8]As the words fewer than 20 times were removed, there are different numbers of words for TripAdvisor and Amazon.

Discussions

This section provides some discussions on the dictionaries constructed for Amazon, in which many of the words are domain-specific. We first take the word *not_waterproof* (the 4th word in $\mathcal{L}_{r_2}$) as an example.

> Disappointed I bought this TV and mounted in my shower so I can watch TV while on the toilet. I went to take a shower later that day and it turns out that they DID *NOT waterproof*!

From the above interesting example, we discover that whether electronic products are waterproof is sometimes an important factor for consumers; obviously, the word *not_waterproof* carries negative sentiment when describing electronic devices. In addition, the strength of the word *really_great* (the 5th word) exceeds that of the word *great* (the 6th word) in $\mathcal{L}_{r_5}$; similarly, the strength of the word *totally_useless* (the 7th word) surpasses that of the word *useless* (the 8th word) in $\mathcal{L}_{r_1}$, which demonstrates that our framework effectively links the intensifying effect of these adverbs to the adjectives and provides more fine-grained dictionaries. This is because that our dictionaries include not only sentiment words but their representations and their degree of relatedness to a certain rating symbol, thereby facilitating further sentiment investigation or applications. Another interesting observation is that the word *not_perfect* is ranked first in $\mathcal{L}_{r_4}$ and the word *not_worst* (the 13th word) is in $\mathcal{L}_{r_2}$. Describing an entity with the original adjective words, *perfect* or *worst*, expresses strong positive or negative sentiments; however, after negated with the function word *not*, the words are shifted to the dictionaries with not-so-strong sentiments. Additionally, the word *not_happy* appears in the top 20 sentiment words of $\mathcal{L}_{r_3}$, $\mathcal{L}_{r_2}$ and $\mathcal{L}_{r_1}$; however, as the rating for the dictionary decreases, its degree of relatedness to the corresponding rating gradually increases, which is 0.283, 0.400, and 0.524, respectively. This result is reasonable as the word *not_happy* apparently carries negative sentiment and the lower the rating of reviews is, the stronger the negative sentiment of words is, which again attests the ability of our framework on producing dictionaries with different granularity.

Conclusions

This paper presents a representation learning framework called "UGSD" for constructing domain-specific sentiment

dictionaries from customer reviews. The learned representations of sentiment words not only allow us to generate opinion dictionaries with different granularity, but extend the application scalability of the constructed lexical words. Both the quantitative evaluation on three datasets and the discussions on the constructed dictionaries show that the framework is effective in constructing high-quality, domain-specific, and fine-grained sentiment dictionaries from customer reviews. The three collected datasets and the source codes are available at https://github.com/cnclabs/UGSD.

Acknowledgments

We thank Chih-Yu Chao for his assistance in partially implementing the program. This research was partially supported by the Ministry of Science and Technology of Taiwan under the grant MOST 107-2218-E-002-061.

References

Baccianella, S.; Esuli, A.; and Sebastiani, F. 2010. Sentiwordnet 3.0: An enhanced lexical resource for sentiment analysis and opinion mining. In *Proceedings of the 7th Conference on International Language Resources and Evaluation*, 2200–2204.

Balahur, A.; Steinberger, R.; Kabadjov, M.; Zavarella, V.; Van Der Goot, E.; Halkia, M.; Pouliquen, B.; and Belyaeva, J. 2010. Sentiment analysis in the news. In *Proceedings of the 7th International Conference on Language Resources and Evaluation*, 2216–2220.

Chao, C.-Y.; Chu, Y.-F.; Yang, H.-W.; Wang, C.-J.; and Tsai, M.-F. 2017. Text embedding for sub-entity ranking from user reviews. In *Proceedings of the 26th ACM International Conference on Information and Knowledge Management*, 2011–2014.

Feldman, R. 2013. Techniques and applications for sentiment analysis. *Communications of the ACM* 56(4):82–89.

Guzman, E., and Maalej, W. 2014. How do users like this feature? a fine grained sentiment analysis of app reviews. In *Proceedings of the 22nd IEEE International Requirements Engineering Conference*, 153–162.

Hamilton, W. L.; Clark, K.; Leskovec, J.; and Jurafsky, D. 2016. Inducing domain-specific sentiment lexicons from unlabeled corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, 595–605.

Hu, M., and Liu, B. 2004. Mining and summarizing customer reviews. In *Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 168–177.

Labille, K.; Gauch, S.; and Alfarhood, S. 2017. Creating domain-specific sentiment lexicons via text mining. In *Proceedings of Workshop on Issues of Sentiment Discovery and Opinion Mining*.

Lu, Y.; Castellanos, M.; Dayal, U.; and Zhai, C. 2011. Automatic construction of a context-aware sentiment lexicon: An optimization approach. In *Proceedings of the 20th International Conference on World Wide Web*, 347–356.

Luca, M., and Zervas, G. 2016. Fake it till you make it: Reputation, competition, and yelp review fraud. *Management Science* 62(12):3412–3427.

Luca, M. 2011. Reviews, reputation, and revenue: The case of yelp.com. *Harvard Business School NOM Unit, Working Paper* (12-016).

Manning, C. D.; Surdeanu, M.; Bauer, J.; Finkel, J.; Bethard, S. J.; and McClosky, D. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, 55–60.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Proceedings of Advances in Neural Information Processing Systems*, 3111–3119.

Pak, A., and Paroubek, P. 2010. Twitter as a corpus for sentiment analysis and opinion mining. In *Proceedings of the 7th Conference on International Language Resources and Evaluation*, 1320–1326.

Reschke, K.; Vogel, A.; and Jurafsky, D. 2013. Generating recommendation dialogs by extracting information from user reviews. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, 499–504.

Ruppenhofer, J.; Brandes, J.; Steiner, P.; and Wiegand, M. 2015. Ordering adverbs by their scaling effect on adjective intensity. In *Proceedings of Recent Advances in Natural Language Processing*, 545–554.

Taboada, M.; Brooke, J.; Tofiloski, M.; Voll, K.; and Stede, M. 2011. Lexicon-based methods for sentiment analysis. *Computational Linguistics* (2):267–307.

Tang, J.; Qu, M.; Wang, M.; Zhang, M.; Yan, J.; and Mei, Q. 2015. LINE: Large-scale Information Network Embedding. In *Proceedings of the 24th International Conference on World Wide Web*, 1067–1077.

Tang, J.; Qu, M.; and Mei, Q. 2015. Pte: Predictive text embedding through large-scale heterogeneous text networks. In *Proceedings of the 21th SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1165–1174.

Wang, C.-J.; Wang, T.-H.; Yang, H.-W.; Chang, B.-S.; and Tsai, M.-F. 2017. Ice: Item concept embedding via textual information. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, 85–94.

Wang, H.; Lu, Y.; and Zhai, C. 2010. Latent aspect rating analysis on review text data: A rating regression approach. In *Proceedings of the 16th SIGKDD International Conference on Knowledge Discovery and Data Mining*, 783–792.

Wiebe, J.; Wilson, T.; and Cardie, C. 2005. Annotating expressions of opinions and emotions in language. *Language Resources and Evaluation* 39(2):165–210.