Community-Based Weighted Graph Model for Valence-Arousal Prediction of Affective Words

Jin Wang, Liang-Chih Yu, *Member, IEEE*, K. Robert Lai, and Xuejie Zhang

***Abstract*—Compared to the categorical approach that represents affective states as several discrete classes (e.g., positive and negative), the dimensional approach represents affective states as continuous numerical values in multiple dimensions, such as the valence-arousal (VA) space, thus allowing for more fine-grained sentiment analysis. In building dimensional sentiment applications, affective lexicons with VA ratings are useful resources but are still very rare. Several semi-supervised methods such as the kernel method, linear regression, and the pagerank algorithm have been investigated to automatically determine the VA ratings of affective words from a set of semantically similar seed words. These methods suffer from two major limitations. First, they apply an equal weight to all seeds similar to an unseen word in predicting its VA ratings. Second, even similar seeds may have quite different ratings (or an inverse polarity) of valence/arousal to the unseen word, thus reducing prediction performance. To overcome these limitations, this study proposes a community-based weighted graph model that can select seeds which are both similar to and have similar ratings (or the same polarity) with each unseen word to form a community (subgraph) so that its VA ratings can be estimated from such high-quality seeds using a weighted propagation scheme. That is, seeds more similar to unseen words contribute more to the estimation process. Experimental results show that the proposed method yields better prediction performance for both English and Chinese datasets.**

***Index Terms*—Affective lexicon, community discovery, sentiment analysis, valence-arousal (VA) prediction, weighted graph model.**

I. Introduction

THANKS to the vigorous development of online social network services, anyone can now easily publish and disseminate articles expressing their thoughts and opinions. Sentiment analysis thus has become a useful technique to automatically identify affective information from texts [1]–[4]. In sentiment analysis, representation of affective states is an essential issue and can be generally divided into categorical and dimensional approaches [5].

Manuscript received October 14, 2015; revised April 20, 2016 and July 11, 2016; accepted July 12, 2016. Date of publication July 27, 2016; date of current version August 12, 2016. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Ani Nenkova. *(Corresponding author: Liang-Chih Yu.)*

J. Wang is with the Department of Computer Science and Engineering, Yuan Ze University, Taoyuan 32003, Taiwan, and also with the School of Information Science and Engineering, Yunnan University, Kunming 650000, China (e-mail: wangjin@ynu.edu.cn).

L. C. Yu is with the Department of Information Management, Yuan Ze University, Taoyuan 32003, Taiwan (e-mail: lcyu@saturn.yzu.edu.tw).

K. R. Lai is with the Department of Computer Science and Engineering, Yuan Ze University, Taoyuan 32003, Taiwan (e-mail: krlai@saturn.yzu.edu.tw).

X. J. Zhang is with the School of Information Science and Engineering, Yunnan University, Kunming 650000, China (e-mail: xjzhang@ynu.edu.cn).

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TASLP.2016.2594287

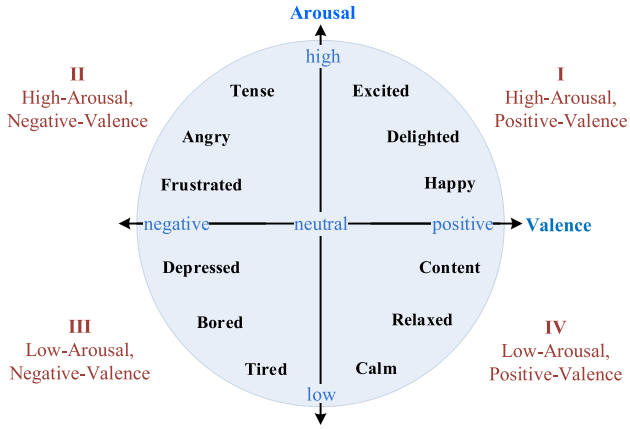


Fig. 1. Two-dimensional VA space.

The categorical approach represents affective states as several discrete classes such as binary (positive and negative) and Ekman's six basic emotions [6] (e.g., anger, happiness, fear, sadness, disgust and surprise). Based on this representation, various techniques have been investigated to develop useful applications such as deceptive opinion spam detection [7], aspect-based sentiment analysis [8]–[10], cross-lingual portability [11], [12], personalized sentiment analysis [13]–[15] and viewpoint identification [16]. In addition to identifying sentiment classes, an extension has been made to further determine their sentiment strength in terms of a multi-point scale [17]–[20].

The dimensional approach represents affective states as continuous numerical values on multiple dimensions, such as valence-arousal (VA) space [21], as shown in Fig. 1. The valence represents the degree of pleasant and unpleasant (or positive and negative) feelings, and the arousal represents the degree of excitement and calm. Based on such a two-dimensional representation, a common research goal is to determine the degrees of valence and arousal of given texts such that any affective state can be represented as a point in the VA coordinate plane. De Choudhury *et al*. used the dimensions of valence and arousal to analyze emotional states (or moods) on Twitter [22]. They suggested that a two-dimensional representation can provide a more fine-grained analysis by capturing the differences in emotional states in both dimensions. For instance, the arousal of the emotional state *depressed* is higher than that of *sad* although they are both negative. Preotiuc-Pietro *et al*. provided a further analysis of Twitter users with a self-reported diagnosis of depression and post-traumatic stress disorder (PSTD), and found that both types of users expressed lower valence and arousal than control (normal) subjects; PTSD users had both higher valence and arousal than depressive users [23]. Recognizing the characteristics of language used by users suffering

from different mental illnesses will help build more intelligent psychological services. For other applications, such as product reviews, presenting customers with high-arousal positive reviews may trigger purchasing behaviors [24]. On the other hand, presenting high-arousal negative reviews for some products (e.g., hedonic products) may also increase sales because they stimulate customer curiosity. However, they point out that current review systems focus on ranking positive and negative reviews. Through dimensional sentiment analysis, both valence and arousal can be used to develop different review ranking strategies for different products.

In developing dimensional sentiment applications, affective lexicons with VA ratings are useful resources but few exist. Most existing applications rely on a handcrafted lexicon ANEW [25] (Affective Norms for English Words) to predict the VA ratings of short and long texts [26]–[29]. ANEW consists of 1,034 English words rated with a 9-point self-assessment manikin (SAM) rating scale [30] in the dimensions of valence, arousal and dominance, where ratings 1 and 9 respectively denote the most negative and positive degrees of affect for valence, and the arousal dimension uses a similar scale to denote calm and excitement. Recently, Warriner *et al*. extended the ANEW to provide a richer lexicon of 13,915 English words [31]. Due to the limited availability of such VA lexicons, especially for Chinese, it is worth developing automatic methods to predict the VA ratings of affective words.

Few studies have sought to predict the VA rating of words using the regression-based method [32], [33] or the graph-based method [34]. These methods usually start from a set of words with labeled VA ratings (called seeds). The VA rating of an unseen word is then estimated from semantically similar seeds. For the regression-based method, Wei *et al*. trained a linear regression model in a cross-lingual manner using the VA ratings of a set of English seed words (source) and their translated Chinese seed words (target) such that the VA ratings can be transformed from a source language to a target language [32]. Malandrakis *et al*. used a kernel function to combine the similarity between seeds and unseen words into a linear regression model [33]. For the graph-based method, Esuli *et al*. transformed WordNet [35] into a graph based on term co-occurrence relations in the glosses of WordNet synsets [34]. The pagerank algorithm was then used to determine the polarity (positive or negative) of WordNet synsets based on both positivity and negativity rankings. In addition to polarity ranking, the pagerank can also be converted for VA prediction.

These methods suffer from two major limitations. First, they estimate the VA ratings of an unseen word by assigning an equal weight to all similar seeds. Second, an unseen word may have many similar seeds, some of which may have quite different ratings (or an inverse polarity) of valence/arousal. To explain this phenomenon, we calculated the similarities of ANEW words using the cosine measure between their corresponding word vectors obtained using the continuous bag-of-words (CBOW) model of word2vec[1] [36], [37]. The cosine measure is chosen because it has been widely used in various applications such as word similarity tasks, semantic/syntactic analogical reasoning, and named entity recognition [36]–[40]. For instance, the five most similar words of the negative word *sad* (1.61)[2] include four negative words i.e., *regretful* (2.82), *terrible* (1.93), *pity* (3.37) and *disgusted* (2.45), and one positive word, *happy* (8.21). This is not surprising because prior studies also reported that even contrasting words (e.g., *happy–sad*) may have similar contexts [41]. The four negative words have the same polarity as *sad* and their ratings are also similar, but the positive word has an inverse polarity and their ratings are quite different. Overall, in ANEW, on average around 70% of the similar words have the same valence polarity for each word, and the remaining 30% have an inverse polarity to that of each word. For the arousal dimension, the ratio of the same and inverse polarity is 6:4 on average. Therefore, an ideal prediction method should account for seeds with the same polarity to an unseen word as they usually have similar VA ratings, and those with an inverse polarity should be excluded as they usually have quite different VA ratings.

Fig. 2 uses a graph representation to explain the idea behind different methods for VA prediction. In Fig. 2(a), the shaded area represents that existing methods [33], [34] estimate the valence rating of the unseen word (e.g., *sad*) by considering all similar words (i.e., those connected to it) including both positive (+) and negative (−) words, and thus may include those with an inverse polarity (e.g., *happy*). A possible way to exclude such noisy words is the use of neighbor selection methods such as k-nearest neighbor (k-NN) [42], ε-nearest neighbor (ε-NN) [43] and graph partition methods such as mincuts [44], [45] and max-flow mincuts [46], [47]. Although these neighbor selection methods have never been used for VA prediction, they can still be applied to this task. The k-NN can be used to exclude noisy words by selecting the top k most similar words as nearest neighbors for an unseen word while ε-NN can select nearest neighbors by introducing a similarity threshold of ε. However, even highly similar words may include those with an inverse polarity to the unseen word. As in the example shown in Fig. 2(b), the noisy word *happy* was not excluded because it was still one of the top k most similar words to *sad* or its similarity exceeded the threshold value ε. Conversely, the useful word *disgusted* was excluded. As shown in Fig. 2(c), for graph partition methods such as mincut or max-flow mincut, the edges with a lower degree of similarity to the unseen word were cut off. This idea is similar to that of k-NN and ε-NN and thus may encounter the same problem.

In this paper, we propose a community-based method for neighbor selection. Unlike ε-NN, k-NN and graph partition methods which select the most similar words as neighbors, the community-based method selects words which are both similar and with the same polarity into a community based on the idea that a word may have more similar neighbors with the same polarity than with an inverse polarity. For example, in Fig. 2(d), the positive word *happy* has more similar neighbors with a positive polarity than negative whereas the negative word *sad* has more

[1] http://code.google.com/p/word2vec/

[2] The number in the parentheses denotes the valence rating of a word. The word with a valence rating greater than 5 can be considered as positive (+), and otherwise negative (–).

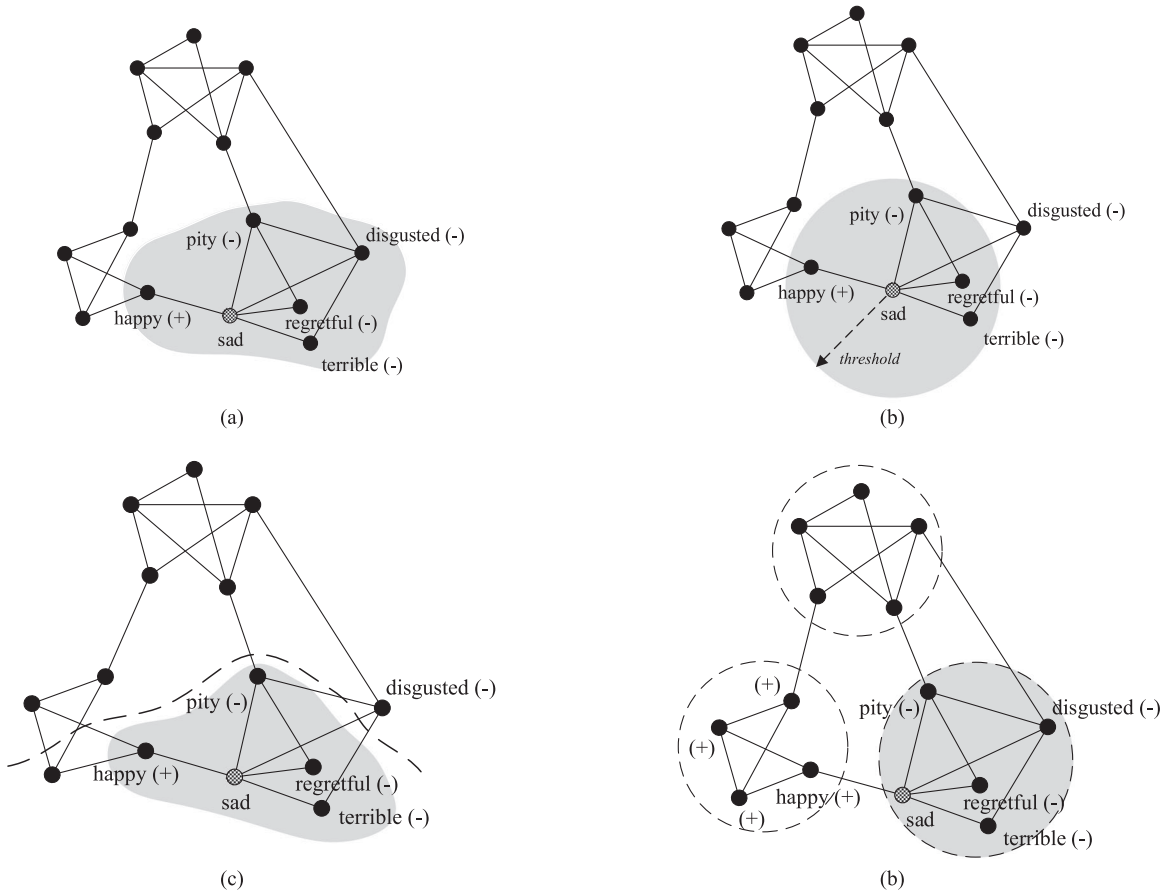


Fig. 2. Illustrative example using different methods for VA prediction. The shaded area represents similar words included in the prediction process. A dashed circle represents a community. (a) existing methods without neighbor selection, (b) k-NN or -NN, (c) graph partition method, (d) community-based method.

with a negative polarity than positive. Hence, these two words with an inverse polarity will not be in the same community even though they are similar. By considering similarity relationships between all words in a graph, the graph can be divided into several communities (sub-graphs) where each community tends to consist of a set of similar words with the same polarity densely connected internally but sparsely connected between different communities. The VA ratings of an unseen word can then be estimated from its neighbors in the community to which it belongs using a weighted propagation mechanism. That is, neighbors more similar to the unseen word may contribute more to the estimation process.

Two sets of experiments were conducted to evaluate the proposed community-based weighted graph model on both English and Chinese affective lexicons with VA ratings. The weighted graph model was first compared against several previously proposed methods such as the linear regression (modified from [32]), kernel method [33], and pagerank [34]. The community-based and other neighbor selection methods were then evaluated to determine whether they could further improve the prediction performance.

The rest of this paper is organized as follows. Section II presents previously proposed methods for VA prediction of affective words. Section III describes the proposed community-based weighted graph model. Section IV summarizes the comparative results of different methods for VA prediction. Conclusions are drawn in Section V.

II. Related Work

Automatically acquiring the VA ratings of affective words is a critical task for building VA lexicons. Existing methods can be generally divided into regression- and graph-based methods.

A. Regression-Based Method

Wei *et al.* proposed a cross-lingual method to transform the VA ratings of English ANEW words to those of Chinese words using linear regression [32]. They first clustered the English words into 66 groups through a suggested upper merged ontology (SUMO) concept. For each cluster, at most three words were randomly selected and translated into their corresponding Chinese words for manual rating. A linear regression model for each cluster was then trained using the VA ratings of both English words (source) and Chinese words (target). This cross-lingual method cannot be directly used in our mono-lingual task because its goal is to transform the VA ratings from one language to another. Therefore, this work modified the cross-lingual linear regression that determines the VA ratings of words in one language from those in another language such that it can determine the VA ratings of words in one language by capturing the

relationship between the similarities and VA ratings among a set of seed words in the same language. For each seed word w_j, the modified linear regression model for valence (or arousal) can be trained using

$$val_{w_i} = b + a \cdot Sim(w_i, w_j) \tag{1}$$

where $Sim(w_i, w_j)$ denotes the similarity between the given seed w_j and another seed w_i; val_{wi} denotes the valence rating of w_i, and both a and b are regression coefficients. The valence model of w_j can be obtained by taking as the input the similarities between w_j and all the other seeds along with their valence ratings, namely $(val_{wi}, Sim(w_i, w_j)|i \neq j$. Similarly, the arousal model of w_j can be built by replacing valence ratings with arousal ratings. Once the regression models for all seeds are obtained, the VA ratings of an unseen word can be predicted using the regression models of the seed word most similar to it by taking their similarity as input.

Malandrakis *et al.* further modified the linear regression model by introducing a kernel function to rescale word similarities [33], that is

$$val_{w_i} = b + \sum_{j=1}^{N_s} a_j \cdot val_{w_j} \cdot f\left[Sim(w_i, w_j)\right] \tag{2}$$

where *f* denotes a kernel function and N_s denotes the number of seed words. Based on this equation, the valence (or arousal) rating of an unseen word w_i can be estimated through a linear combination of the valence (or arousal) ratings of its similar seeds w_j and their similarities.

Astudillo *et al*. proposed a regression method directly trained on word embedding vectors to determine intensity scores for Twitter terms with a positive sentiment [48]. Tang *et al.* also presented a representation learning method to identify the polarity of Twitter terms [49].

B. Graph-Based Method

Graph-based methods have been used to determine the polarity of words by applying label propagation [50], [51] and pagerank [34] on a graph. For the label propagation, the polarity of an unseen word was determined by propagating the polarity labels from its neighbors. In addition to polarity identification (discrete), the pagerank can further handle continuous values for polarity ranking [34]. We thus converted the pagerank algorithm for VA prediction by taking the ranking scores output by the pagerank as VA ratings, as shown in Eq. (3).

$$val^t_{w_i} = \alpha \sum_{w_j \in Nei(w_i)} \frac{val^{t-1}_{w_j}}{|Nei(w_i)|} + (1-\alpha)e \tag{3}$$

where w_i and w_j respectively denote an unseen word and a seed word; $Nei(w_i)$ denotes the neighbor nodes of w_i (i.e., the seeds similar to w_i); val^t_{wi} denotes the predicted rating of valence of w_i at the *t*th iteration; *e* is a constant, and α is a decay factor parameter. Based on Eq. (3), the VA ratings of the unseen words were iteratively estimated from all of its similar neighbors and each was assigned an equal weight.

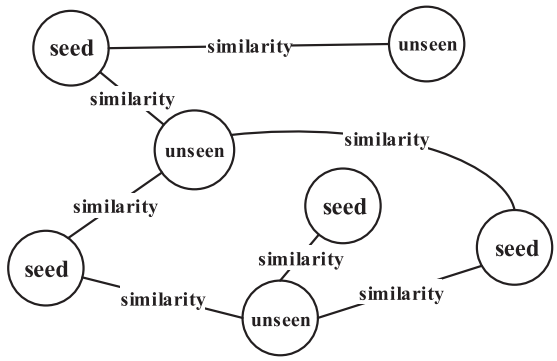


Fig. 3. Conceptual diagram of a weighted graph model for VA prediction.

III. Community-Based Weighted Graph Model

The procedure for using the community-based weighted graph model for VA prediction is described as follows. Given an unseen word and a set of seed words with labeled VA ratings, their similarities are first calculated to build a weighted graph model where each node represents a word and each edge represents the similarity between two nodes. To exclude the noisy neighbors linked to the unseen word, a community detection method is used to select useful neighbors into a community. Finally, the VA ratings of the unseen word are estimated from its community members using a weighted propagation mechanism. The following sub-sections explain the details of the similarity calculation, weighted graph model and community-based neighbor selection.

A. Similarity Calculation

While traditional vector space models provide a firm basis to capture distributional semantics [52], [53], continuous vector representations of words have attracted significant attention in recent years [54], [55]. Many algorithms and tools thus have been developed to learn continuous vector representation of words such as word2vec (CBOW and skip-gram models) [36], [37], GloVe [38], and Scratch [56]. These algorithms can be trained on large amounts of text data, so that the models can capture substantial amounts of semantic information. This study builds word vectors using the CBOW model of word2vec. Once the vectors of each word are obtained, a cosine distance is applied to measure the semantic similarity between words.

B. Weighted Graph Model

Based on the theory of link analysis, the relations between unseen words and seed words can be considered as a graph, as shown in Fig. 3. The VA ratings of each unseen word can then be predicted through the links connected to the seed words to which it is similar using their similarities as weights.

The formal definition of a graph model is described as follows. Let $G = (\mathrm{V}, \mathrm{E})$ be an undirected graph, where *V* denotes a set of words and *E* denotes a set of undirected edges. Each edge *e* in *E* denotes a relation between word w_i and word w_j in $V(1 \leq i, j \leq n, i \neq j)$, representing the similarity between them. For each node $w_i, Nei(w_i) = w_j|(w_j, w_i)E$ denotes the

set of its neighbor nodes, representing a set of words to which it is similar (i.e., those with a similarity greater than zero). The valence or arousal of w_i, denoted as val_{wi} or aro_{wi}, can then be iteratively determined by its neighbors, defined as, as eq. (4) shown in bottom of this page.

where t denotes the tth iteration and α is a decay factor or a confidence level for computation (a constant between 0 and 1), which limits the effect of rank sinks to guarantee convergence to a unique vector. Initially ($t = 0$), and the constant value of valence and arousal of each unseen word is assigned as 5 because the VA values are predicted in the range of 1 to 9. Later ($t > 0$), it is iteratively updated through graph propagation until convergence.

Equation (4) selects an unseen word each time. If the input contains multiple unseen words, then it can be transformed into a matrix notation to handle all unseen words at a time. Suppose that the vectors,

$$\mathbf{V} = (val_{w_1}, val_{w_2}, \ldots, val_{w_N})^T,$$
$$\mathbf{A} = (aro_{w_1}, aro_{w_2}, \ldots, aro_{w_N})^T$$

are the vectors of the VA rating of all words N (including seed words and unseen words). Matrix

$$\mathbf{S} = \begin{bmatrix} Sim(w_1,w_1) & \cdots & Sim(w_1,w_j) & \cdots & Sim(w_1,w_N) \\ \vdots & & \vdots & & \vdots \\ Sim(w_i,w_1) & \cdots & Sim(w_i,w_j) & \cdots & Sim(w_i,w_N) \\ \vdots & & \vdots & & \vdots \\ Sim(w_N,w_1) & \cdots & Sim(w_N,w_j) & \cdots & Sim(w_N,w_N) \end{bmatrix}$$

is the adjacency matrix of each words, where $Sim(\mathrm{w_i}, \mathrm{w_j})$ represents the similarity between words i and j, where $i, \mathrm{j} = 1, 2, \ldots, n$, $\mathrm{i} \neq \mathrm{j}$.

Given other two vectors $\mathbf{I} = (1, 1, \ldots, 1)^{\mathrm{T}}$ and $\mathbf{D} = (d_1, d_2, \ldots, d_N)^T$, where

$$d_i = \begin{cases} \alpha & \text{if} \quad w_i \in \textbf{unseen} \\ 0 & \text{if} \quad w_i \in \textbf{seed} \end{cases},$$

where α is the previously mentioned decay factor. For vectors $\mathbf{A} = (\mathrm{a_1}, \mathrm{a_2}, \ldots, \mathrm{a_n})^{\mathrm{T}}$ and $\mathbf{B} = (\mathrm{b_1}, \mathrm{b_2}, \ldots, \mathrm{b_n})^{\mathrm{T}}$, function $\mathcal{M}(\mathbf{A}, \mathbf{B})$ and $\mathcal{D}(\mathbf{A}, \mathbf{B})$ can be defined as,

$$\mathcal{M}(\mathbf{A}, \mathbf{B}) = (a_1 \times b_1, a_2 \times b_2, \ldots, a_n \times b_n)^T,$$
$$\mathcal{D}(\mathbf{A}, \mathbf{B}) = (a_1/b_1, a_2/b_2, \ldots, a_n/b_n)^T.$$

Then, Eq. (4) can be turned into the following matrix format,

$$\begin{aligned} \mathbf{V}_t &= \mathcal{M}[(\mathbf{I} - \mathbf{D})^T, \mathbf{V}_{t-1}] + \mathcal{M}[\mathbf{D}^T, \mathcal{D}(\mathbf{S}\mathbf{V}_{t-1}, \mathbf{S} \cdot \mathbf{I})] \\ \mathbf{A}_t &= \mathcal{M}[(\mathbf{I} - \mathbf{D})^T, \mathbf{A}_{t-1}] + \mathcal{M}[\mathbf{D}^T, \mathcal{D}(\mathbf{S}\mathbf{A}_{t-1}, \mathbf{S} \cdot \mathbf{I})]. \end{aligned} \quad (5)$$

C. Community-Based Neighbor Selection

The weighted graph model estimates the VA ratings of an unseen word by considering all similar neighbors, and thus may include some noisy neighbors with an inverse polarity of valence/arousal to that of the unseen word. To tackle this problem, we develop a community detection method to select similar neighbors with the same polarity into the same community. This can be accomplished based on the notion that a word may have more similar neighbors with the same polarity than with an inverse polarity. That is, a group of similar words with dense connections within the group and sparse connections between different groups would be a good candidate of communities. Therefore, we introduce a modularity value [57], [58] to measure the associations within and between communities over the graph such that it can be partitioned into several non-overlapping communities (sub-graphs), i.e. $G = (V, E) = C_1 \cup C_2 \cup \ldots \cup C_n | C_i \cap C_j = \varnothing, 1 \leq i, j \leq n, i \neq j$. The modularity M is defined as

$$M = \sum_C \left[\frac{Sim_{\mathrm{within},C}}{2m} - \left(\frac{Sim_{\mathrm{between},C}}{2m} \right)^2 \right], \quad (6)$$

where $Sim_{\mathrm{within},C}$ denotes the sum of the weights (similarities) between all words within a community C, that is

$$Sim_{\mathrm{within},C} = \sum_{w_i \in C} \sum_{w_j \in C} Sim(w_i, w_j) \quad (7)$$

$Sim_{\mathrm{between},C}$ denotes the sum of weights between each word in C and all the other words in the graph, that is

$$Sim_{\mathrm{between},C} = \sum_{w_i \in C} \sum_{w_j \in G} Sim(w_i, w_j), \quad (8)$$

and $2m$ denotes the total weights between all words in the graph, that is

$$2m = \sum_{w_i, w_j \in G} Sim(w_i, w_j). \quad (9)$$

Based on this modularity, community detection searches for the partition that maximizes the modularity over the graph. This can be accomplished by iteratively repeating the modularity optimization and community merge steps as follows.

1) Modularity Optimization Step: Initially, each word in the graph is assigned to its own distinct community. Each word is then sequentially moved from the original community to all its neighbor communities. Each movement of a word will lead to a change of modularity ΔM, defined as,

$$\Delta M = \Delta M_{\mathrm{move\_out}} + \Delta M_{\mathrm{move\_in}}, \quad (10)$$

where $\Delta \mathrm{M_{move\_out}}$ and $\Delta \mathrm{M_{move_in}}$ respectively denote the change of modularity caused by moving a word w_i away from its original community C_i and into a new community C_j. The

$$val_{w_i}^t = \begin{cases} \text{constant} & (t = 0) \\ (1 - \alpha) \cdot val_{w_i}^{t-1} + \alpha \dfrac{\sum_{w_j \in Nei(w_i)} Sim(w_i, w_j) \cdot val_{w_j}^{t-1}}{\sum_{w_j \in Nei(w_i)} Sim(w_i, w_j)} & (t > 0) \end{cases} \quad (4)$$

$\Delta \mathrm{M}_{\mathrm{move_out}}$ is calculated as

$$\begin{aligned}\Delta M_{\mathrm{move\_out}} &= M^{\mathrm{after}}_{\mathrm{move\_out}} - M^{\mathrm{before}}_{\mathrm{move\_out}} \\ &= \left[\frac{Sim_{\mathrm{within},C_i}}{2m} - \left(\frac{Sim_{\mathrm{between},C_i}}{2m}\right)^2 - \left(\frac{k_{w_i}}{2m}\right)^2\right] \\ &- \left[\frac{Sim_{\mathrm{within},C_i} + k_{w_i,C_i}}{2m} - \left(\frac{Sim_{\mathrm{between},C_i} + k_{w_i}}{2m}\right)^2\right],\end{aligned} \tag{11}$$

where $M^{\mathrm{after}}_{\mathrm{move\_out}}$ and $M^{\mathrm{before}}_{\mathrm{move\_out}}$ respectively denote the modularity after and before moving w_i away from C_i. Both are calculated based on Eq. (6). Fig. 4 illustrates the process of moving w_i from C_i to C_j. In Fig. 4(a), the dashed lines denote the edges connected between w_i and the words in C_i, and the sum of their weights is denoted as $k_{w_i,C_i} = \sum_{w_j \in C_i} Sim(w_i, w_j)$ in $M^{\mathrm{before}}_{\mathrm{move\_out}}$. This term will be deleted in $M^{\mathrm{after}}_{\mathrm{move\_out}}$ because the edges within C_i connected between w_i and the words in C_i will be removed once w_i is moved away from C_i. Similarly, moving w_i away from C_i will also decrease the weight between C_i, and thus $k_{w_i} = \sum_{w_j \in G} Sim(w_i, w_j)$ in $M^{\mathrm{before}}_{\mathrm{move\_out}}$ is deleted in $M^{\mathrm{after}}_{\mathrm{move\_out}}$. The term $-(\frac{k_{w_i}}{2m})^2$ appears in $M^{\mathrm{after}}_{\mathrm{move\_out}}$ because after w_i leaves C_i, itself will temporarily form a community before entering C_j, as the dashed circle shown in Fig. 4(b). Therefore, in addition to the modularity of C_i, the modularity of the temporary community of w_i is added in $M^{\mathrm{after}}_{\mathrm{move\_out}}$. It is also calculated based on Eq. (6); that is $M = \frac{Sim_{\mathrm{within},C}}{2m} - (\frac{Sim_{\mathrm{between},C}}{2m})^2$, where $Sim_{\mathrm{within},C} = 0$ because in this temporary stage w_i itself is a community, and $Sim_{\mathrm{between},C} = k_{w_i}$, thus yielding $M = 0 - (\frac{k_{w_i}}{2m})^2 = -(\frac{k_{w_i}}{2m})^2$.

Similar to $\Delta \mathrm{M}_{\mathrm{move_out}}$, $\Delta \mathrm{M}_{\mathrm{move_in}}$ can be defined as

$$\begin{aligned}\Delta M_{\mathrm{move\_in}} &= M^{\mathrm{after}}_{\mathrm{move\_in}} - M^{\mathrm{before}}_{\mathrm{move\_in}} \\ &= \left[\frac{Sim_{\mathrm{within},C_j} + k_{w_i,C_j}}{2m} - \left(\frac{Sim_{\mathrm{between},C_j} + k_{w_i}}{2m}\right)^2\right] \\ &- \left[\frac{Sim_{\mathrm{within},C_j}}{2m} - \left(\frac{Sim_{\mathrm{between},C_j}}{2m}\right)^2 - \left(\frac{k_{w_i}}{2m}\right)^2\right].\end{aligned} \tag{12}$$

For $M^{\mathrm{before}}_{\mathrm{move\_in}}$, the modularity calculation is similar to $M^{\mathrm{after}}_{\mathrm{move\_out}}$; that is it considers both the modularity of C_j and temporary community of w_i. For $M^{\mathrm{after}}_{\mathrm{move\_in}}$, moving w_i into C_j will introduce new edges into C_j; that is, those connected between w_i and the words in C_j, as the dashed lines shown in Fig. 4(c). Hence, the weight within C_j ($Sim_{\mathrm{within},C_j}$) is added by $k_{w_i,C_j} = \sum_{w_j \in C_j} Sim(w_i, w_j)$, defined as the sum of the weights between the newly moved-in word w_i and the words in C_j. Similarly, moving w_i into C_j will also increase the weight between C_j, and thus $Sim_{\mathrm{between},C_j}$ is added by $k_{w_i} = \sum_{w_j \in G} Sim(w_i, w_j)$, defined as the sum of the weights between w_i and all words in the graph. On the other hand, the temporary term $-(\frac{k_{w_i}}{2m})^2$ appears in both $M^{\mathrm{after}}_{\mathrm{move\_out}}$ and $M^{\mathrm{before}}_{\mathrm{move\_in}}$ will be canceled out in calculating ΔM.

After trying to move w_i into all its neighbor communities, the movement yielding the highest $\Delta \mathrm{M}$ will be taken, and only if $\Delta \mathrm{M}$ is positive. Otherwise, w_i will stay in the original community. The movement procedure is performed sequentially and repeatedly for all words in the graph until no positive $\Delta \mathrm{M}$ is found for all movements.

2) Community Merge Step: Based on the communities found in the previous step, this step treats them as new nodes to build a new weighted graph. The weight of each edge between two nodes (communities) is calculated by the sum of the weights between all words in the two communities. In addition, two communities are considered neighbor nodes if they have at least one edge between them. The new graph is then passed back to the previous modularity optimization step. These two steps are iteratively performed until no more new communities are found. The graph is then partitioned into several communities. The optimal number of communities will be empirically determined using a development set.

In making predictions, both unseen word(s) and seeds are used for community discovery by iteratively repeating the above modularity optimization and community merge steps until the optimal number of communities is reached. The VA ratings of each unseen word are then predicted through the neighbors in the community to which it belongs using the weighted graph model, while those in different communities are ignored so as to exclude noisy neighbors.

IV. Experimental Results

This section presents the experimental results of the community-based weighted graph model for VA prediction on both English and Chinese affective lexicons. We first compare the weighted graph model against several previously proposed methods. The community-based and other neighbor selection methods are then evaluated on the weighted graph model to examine the effect of noisy word removal on performance improvement.

A. Experiment Settings

Datasets: This experiment used two affective lexicons with VA ratings: i) ANEW which contains 1,034 English affective words [25] and ii) Chinese valence-arousal words (CVAW) taken from Chinese linguistic inquiry and word count (C-LIWC) [59], which contains 1,653 words with manually rated VA values using the SAM scale of 1 to 9 [30]. Each lexicon was randomly split into a training set, development set and test set using a 6:2:2 ratio for 5-fold cross-validation. For each run, the training set was used as seeds, the development set was used for parameter selection for neighbor selection methods (see Table III), and the test set was used for performance evaluation. The similarities between English words and Chinese words were calculated using the CBOW model of word2vec (dimensionality = 300, window size = 5) trained with the respective English and Chinese wiki corpora[3].

Evaluation Metrics: Prediction performance was evaluated by examining the difference between the predicted values of VA

[3]https://dumps.wikimedia.org/

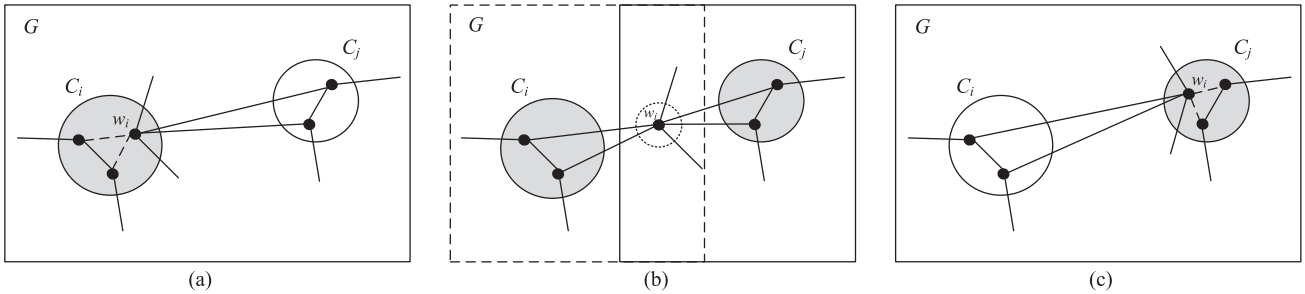


Fig. 4. Illustration of word movement between communities. (a) Before_Move_out, (b) After_Move_out Before_Move_in, (c) After_Move_in

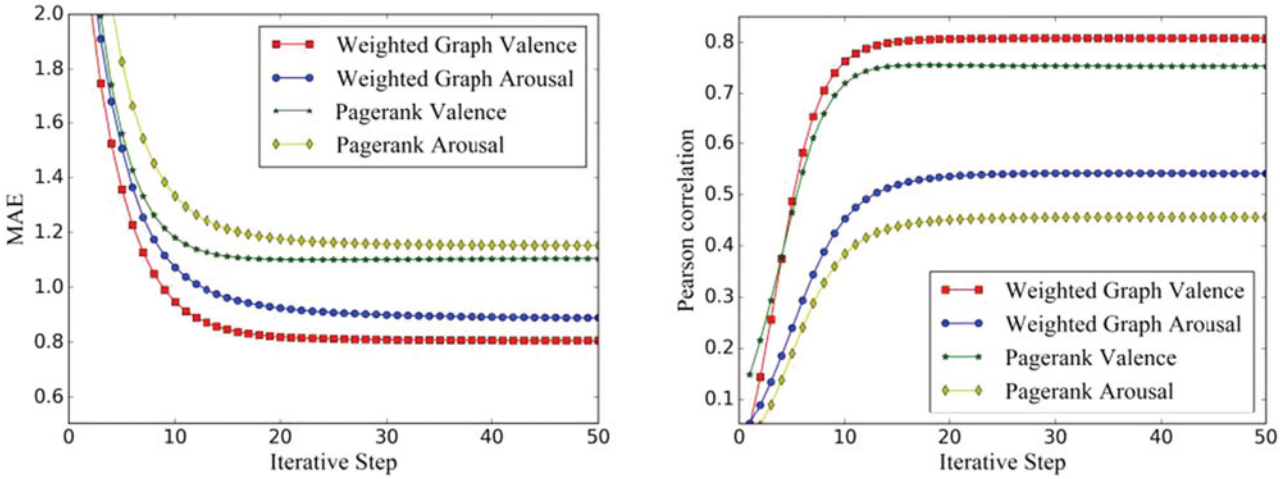


Fig. 5. Iterative results of the pagerank algorithm and weighted graph model.

ratings and the corresponding actual values in the ANEW and CVAW lexicons. The evaluation metrics included:

a) Mean Absolute Error (MAE):

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|A_i - P_i| \tag{13}$$

b) Pearson Correlation Coefficient (r):

$$r = \frac{1}{n-1}\sum_{i=1}^{n}\left(\frac{A_i - \bar{A}}{\sigma_A}\right)\left(\frac{P_i - \bar{P}}{\sigma_P}\right) \tag{14}$$

where A_i is the actual value, P_i is the predicted value, n is the number of test samples, $\bar{A}$ and $\bar{P}$ respectively denote the arithmetic mean of A and P, and σ is the standard deviation. The MAE measures the error rate and the Pearson correlation coefficient (r) measures the linear correlation between the actual values and the predicted values. A lower MAE and a higher r indicate more accurate prediction performance.

B. Evaluation of Weighted Graph Model

This section evaluates regression- and graph-based methods without neighbor selection for VA prediction. The implementation details for each method are described as follows.

1) *LR(sim):* This method was implemented using Eq. (1), which was modified from [32]. It trained a linear regression model for valence/arousal for each seed in the training set by taking as the input the similarities between the seed and the other seeds in the training set and their valence/arousal ratings. In testing, the VA ratings of each unseen word were determined using the models of the seed to which is was most similar by taking their similarity as input.

2) *LR(vec):* Instead of using similarities for training, a variant of linear regression was implemented by training directly on the word embedding vectors. That is,

$$val_{w_i} = b + a \cdot \mathrm{wordvec}(w_i) \tag{15}$$

where *wordvec*(w_i) denotes the word vector of a word w_i obtained using word2vec. Once the word vectors of all seeds in the training set were obtained, they were used for training along with their valence/arousal ratings. The trained models were then used to predict the VA ratings of each unseen word based on its word vector.

1) *Kernel:* This method was implemented using Eq. (2). It was trained by considering all seeds in the training set as training instances. For each training instance, the similarities of all its similar words (similarity > 0) and their VA ratings were used for training. In addition, for the kernel method, the linear similarity function was chosen because it yielded top performance in the previous study [33]. The trained models were then used to predict the VA ratings of each unseen word based on the VA ratings and similarities of all its similar seeds (similarity > 0).

2) *PageRank:* This method was implemented based on [34]. A graph was first constructed using both the seeds in

TABLE I
COMPARATIVE RESULTS OF DIFFERENT METHODS FOR VA PREDICTION

Valence	ANEW (English)		CVAW (Chinese)	
	MAE	r	MAE	r
Kernel	1.368	0.601	1.347	0.618
LR(sim)	1.342	0.622	1.312	0.629
LR(vec)	1.219	0.734	1.154	0.737
PageRank	1.113	0.749	1.130	0.719
Weighted Graph	0.817	0.801	0.897	0.782
Arousal	ANEW (English)		CVAW (Chinese)	
	MAE	r	MAE	r
Kernel	1.354	0.418	1.352	0.414
Linear Regression(sim)	1.348	0.426	1.336	0.423
Linear Regression(vec)	0.989	0.465	1.028	0.465
PageRank	1.124	0.450	1.138	0.461
Weighted Graph	0.881	0.539	0.913	0.542

the training set and unseen word(s). The VA ratings of the unseen word(s) were estimated by assigning an equal weight to the edges connected to its similar seeds (similarity > 0).

3) *Weighted Graph:* This method was implemented using Eqs. (4) and (5). A weighted graph was constructed using both the seeds in the training set and the unseen word(s) by considering their similarities as weights. The VA ratings of the unseen word(s) were estimated by from its similar seeds (similarity > 0) according to their weights.

1) Iterative Results of Graph-Based Methods: Since both the graph-based methods (PageRank and Weighted Graph) used an iterative procedure for VA prediction, Fig. 5 shows their iterative results using the MAE and Pearson correlation coefficient. The results show that the performance of both methods stabilized after around 10 iterations, indicating its efficiency for VA prediction. Another observation is that the ultimate converging result of each word is unrelated to the decay factor and the initial random assignment.

2) Comparative Results: Table I compares the results of the regression-based methods (LR(sim), LR(vec) and Kernel), and graph-based methods (PageRank and Weighted Graph). The performances (MAE and r) of PageRank and Weighted Graph were taken from results of the 50th iteration. The results show that the proposed Weighted Graph yielded a smaller error rate[4] and a higher correlation level than the other methods on both the ANEW and CVAW lexicons. The weighted graph model achieved better performance because it predicted VA ratings by considering both the relations of multiple nodes and the weights between them. For the regression-based methods, LR(vec) directly trained on word vectors achieved best performance, while both Kernel and LR(sim) trained on word similarities achieved similar results. In addition, both graph-based methods outperformed the regression-based methods, except for LR(vec) on the arousal dimension. Another observation is that the correlations for arousal prediction were smaller than those for valence prediction, indicating that the arousal dimension is more difficult to predict.

3) Error Analysis: Although the weighted graph model achieved best performance, it may still include noisy words (neighbors) because it considered all similar words in the prediction process just like the other methods do. Table II lists some example words with largest errors predicted by the weighted graph model, where the last column shows the 10 most similar neighbors for each example word along with their valence or arousal ratings (presented in parenthesis) taken from ANEW. The results show that the 10 most similar neighbors for each example word contain several noisy neighbors (e.g., *selfish* (2.42) and *greedy* (3.51) were noisy neighbors for *wealthy* (7.70)), and the VA ratings of the noisy neighbors were quite different to those of their corresponding example words.

C. Evaluation of Community-Based Neighbor Selection

To investigate the effect of excluding noisy words from prediction processing, several neighbor selection methods such as k-NN [42], ε-NN [43], mincuts [44], [45], max-flow mincuts [46], [47] and the proposed community detection method were evaluated to determine whether they can further improve the performance of VA prediction. The k-NN was implemented by selecting top k most similar words as the nearest neighbors for each unseen word, while ε-NN was implemented by selecting those with similarity between each word exceeding a predefined threshold value ε. Mincuts removed edges with the lowest degree of similarity and divided the graph into several connected parts. Max-flow mincuts is an improvement of mincuts by introducing a max-flow algorithm, which produces only one of several possible mincuts. Both were implemented using the NetworkX[5] toolkit [60]. The proposed community detection method was implemented using Eq. (6) and Eq. (10) to iteratively group a set of similar words with the same polarity or similar VA ratings into the same community. The nearest neighbors derived from the above neighbor selection methods were then used for VA prediction using the regression- and graph-based methods presented in the previous section.

For LR(sim), only the nearest neighbors, rather than all seeds, were used to train the valence/arousal models for each seed, and the models of the seed most similar to each unseen word were used for VA prediction. For LR(vec), the word vectors of the nearest neighbors to each unseen word were used for training and then testing based on the word vector of the unseen word. For Kernel, the nearest neighbors of each seed were used for training, and those of each unseen word were used for VA prediction. For both PageRank and Weight Graph, the VA ratings of each unseen word were determined from its nearest neighbors based on a graph.

1) Selection of Optimal Parameters: Different neighbor selection methods may have different optimal parameter settings. Fig. 6 shows the optimal settings of each neighbor selection method for the weighted graph model, derived from an ANEW development set using MAE as an example metric. For k-NN,

[4]Another metric root mean squared error (RMSE) is also tested and presents similar trends.

[5]https://networkx.github.io/

TABLE II
ERROR ANALYSIS OF WORDS WITH THE GREATEST ABSOLUTE ERROR PREDICTED BY THE WEIGHTED GRAPH MODEL

Valence	Actual value	Predicted value	Error	Top 10 most similar neighbors
paradise	8.72	6.70	2.02	heaven (7.30), bliss (6.95), beautiful (7.60), hell (2.24), dream (6.73), swamp (5.14), lonely (2.17), carefree (7.54), nightmare (1.91)
wealthy	7.70	5.81	1.89	millionaire (8.03), luxury (7.88), handsome (7.93), lavish (6.21), greed (3.51), riches (7.70), famous (6.98), money (7.59), modest (5.76), selfish (2.42)
funeral	1.39	3.18	1.79	burial (2.05), cemetery (2.63), coffin (2.56), wedding (7.82), morgue (1.92), grief (1.69), church (6.28), family (7.65), tomb (2.94), bereavement (4.57)
sad	1.61	3.48	1.87	regretful (2.82), terrible (1.93), happy (8.21), pity (3.37), disgusted (2.45), thankful (6.89), lonely (2.17), grateful (7.37), cruel (1.97), stupid (2.31)
Arousal	**Actual value**	**Predicted value**	**Error**	**Top 10 most similar neighbors**
enraged	7.97	6.15	1.81	angry (7.17), disgusted (5.42), frustrated (5.61), displeased (5.64), unhappy (4.18), resent (4.47), startled (6.93), terrified (7.83), upset (5.86), astonished (6.58)
ambulance	7.33	5.46	1.87	hospital (5.98), taxi (3.41), bus (3.55), nurse (4.84), truck (4.84), trauma (6.33), doctor (5.86), morgue (4.84), accident (6.26), vehicle(4.63)
bored	2.83	4.59	1.76	frustrated (5.61), lazy (2.65), addicted (4.81), fatigued (2.64), confused (6.03)mad (6.76), lonely (4.51), seasick (5.80), scared (6.82), discouraged (4.53)
peace	2.95	4.68	1.73	justice (5.47), freedom (5.52), liberty (5.60), war (7.49), life (6.02), bless (4.05), dignified (4.12), disturb (5.80), hope (5.44), mind (5.00)

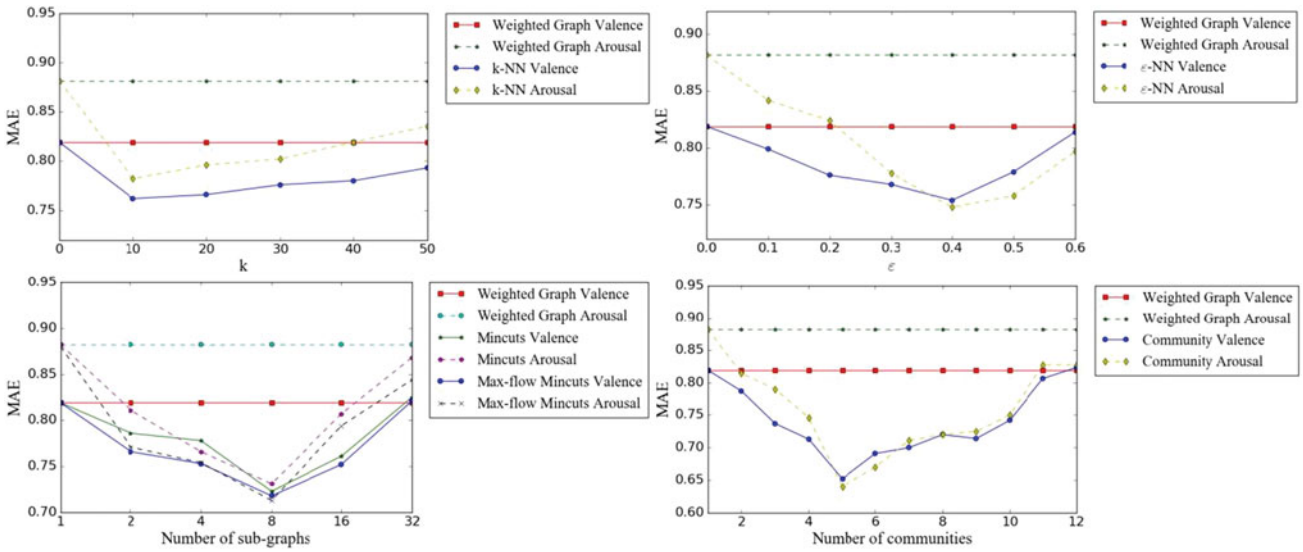


Fig. 6. Parameter selection for different neighbor selection methods for the weighted graph model, evaluated on an ANEW development set using MAE.

selecting the 10 most similar neighbors to predict the VA ratings of each unseen word yielded the lowest MAE. For ε-NN, neighbors with a similarity between the unseen words greater than 0.4 were included in the prediction process. For both mincuts and max-flow mincuts, the optimal setting was obtained by dividing the graph into 8 sub-graphs. For the community-based method, the optimal number of communities was 5. The optimal settings for CVAW were $k = 10, \varepsilon = 0.4$, and both the number of sub-graphs and communities $= 8$. Too few or too many communities did not achieve good results because increasing the number of communities will decrease the number of words in each community, thus potentially including noisy words or excluding useful words for prediction. Table III summarizes the optimal settings of each neighbor selection method for all VA prediction methods.

2) Comparative Results: Tables IV and V present the results of different neighbor selection methods for regression- and graph-based methods on ANEW and CVAW, respectively.

TABLE III
OPTIMIZED PARAMETER SETTINGS OF NEIGHBOR SELECTION METHODS

ANEW	LR (sim)	LR (vec)	Kernel	PageRank	Weighted Graph
k-NN	50	40	50	10	10
ε-NN	0.2	0.3	0.3	0.4	0.4
mincuts	8	8	8	8	8
max-flow mincuts	8	8	8	8	8
community	5	6	5	5	5

The results show that all neighbor selection methods did improve the performance of VA prediction. Overall, both *k*-NN and ε-NN yielded similar results. Both graph partition methods (mincuts and max-flow mincuts) also yielded similar results, and their performances were better than those of both *k*-NN and ε-NN. The community-based method achieved best performance among all the neighbor selection methods.

TABLE IV
COMPARATIVE RESULTS OF DIFFERENT NEIGHBOR SELECTION METHODS FOR VA PREDICTION (ANEW)

Valence	Inverse Polarity	MAE					r				
		LR (sim)	Kernel	Page Rank	LR (vec)	Weight Graph	LR (sim)	Kernel	Page Rank	LR (vec)	Weight Graph
No Neighbor Selection	29.38%	1.342	1.368	1.113	1.219	0.817	0.622	0.601	0.749	0.734	0.801
k-NN	22.88%	1.263	1.255	1.019	0.934	0.762	0.634	0.734	0.764	0.778	0.818
ε-NN	22.35%	1.279	1.289	1.009	0.989	0.754	0.626	0.723	0.769	0.753	0.821
Mincuts	20.55%	1.151	1.065	0.899	0.837	0.723	0.649	0.751	0.791	0.799	0.832
Max-flow mincuts	20.39%	1.138	1.042	0.892	0.841	0.718	0.661	0.756	0.794	0.794	0.837
Community	11.70%	**1.083**	**0.845**	**0.792**	**0.727**	**0.651**	**0.708**	**0.786**	**0.814**	**0.826**	**0.908**
Arousal	Inverse Polarity	MAE					r				
		LR (sim)	Kernel	Page Rank	LR (vec)	Weight Graph	LR (sim)	Kernel	Page Rank	LR (vec)	Weight Graph
No Neighbor Selection	39.85%	1.348	1.354	1.124	0.989	0.881	0.426	0.418	0.450	0.465	0.539
k-NN	30.45%	1.185	1.165	1.062	0.890	0.781	0.462	0.444	0.484	0.512	0.564
ε-NN	31.02%	1.158	1.192	1.045	0.917	0.748	0.480	0.458	0.488	0.499	0.576
Mincuts	27.92%	1.071	0.961	0.909	0.813	0.731	0.508	0.512	0.517	0.549	0.579
Max-flow mincuts	28.16%	1.023	0.992	0.892	0.823	0.713	0.514	0.501	0.518	0.538	0.583
Community	22.34%	**0.969**	**0.827**	**0.801**	**0.752**	**0.638**	**0.535**	**0.542**	**0.558**	**0.585**	**0.678**

TABLE V
COMPARATIVE RESULTS OF DIFFERENT NEIGHBOR SELECTION METHODS FOR VA PREDICTION (CVAW)

Valence	Inverse Polarity	MAE					r				
		LR (sim)	Kernel	Page Rank	LR (vec)	Weight Graph	LR (sim)	Kernel	Page Rank	LR (vec)	Weight Graph
No Neighbor Selection	28.48%	1.312	1.347	1.130	1.154	0.897	0.629	0.618	0.719	0.737	0.782
k-NN	22.01%	1.273	1.270	1.067	0.951	0.848	0.639	0.650	0.765	0.764	0.819
ε-NN	21.71%	1.278	1.274	1.078	0.962	0.842	0.638	0.645	0.769	0.754	0.822
Mincuts	20.19%	1.176	1.045	0.855	0.867	0.833	0.678	0.728	0.799	0.804	0.831
Max-flow mincuts	20.41%	1.152	1.017	0.854	0.874	0.828	0.681	0.734	0.794	0.791	0.838
Community	11.12%	**1.014**	**0.917**	**0.828**	**0.798**	**0.781**	**0.725**	**0.768**	**0.813**	**0.855**	**0.886**
Arousal	Inverse Polarity	MAE					r				
		LR (sim)	Kernel	Page Rank	LR (vec)	Weight Graph	LR (sim)	Kernel	Page Rank	LR (vec)	Weight Graph
No Neighbor Selection	34.72%	1.336	1.352	1.138	1.028	0.913	0.423	0.414	0.461	0.465	0.542
k-NN	31.54%	1.182	1.170	1.078	0.868	0.842	0.462	0.457	0.499	0.525	0.546
ε-NN	30.26%	1.176	1.174	1.068	0.894	0.819	0.475	0.453	0.504	0.493	0.557
Mincuts	29.36%	1.005	1.045	0.863	0.816	0.742	0.497	0.508	0.524	0.552	0.586
Max-flow mincuts	29.28%	1.003	1.072	0.845	0.807	0.722	0.503	0.494	0.527	0.558	0.595
Community	21.15%	**0.879**	**0.934**	**0.785**	**0.746**	**0.616**	**0.543**	**0.523**	**0.557**	**0.596**	**0.694**

To further explain the effectiveness of neighbor selection, we calculated the percentages of neighbors with an inverse polarity of valence/arousal to that of the unseen words included in the prediction process, as shown in column "Inverse Polarity". These percentages were averaged over all VA prediction methods. Without neighbor selection, around 29% neighbors in both ANEW and CVAW with an inverse polarity to that of the unseen words were included for valence prediction, and around 35% to 40% for arousal prediction. Predicting the VA ratings of unseen words from all their similar words may include such noisy words and resulted in reduced performance because the noisy words may have quite different VA ratings from the unseen words. Once the neighbor selection methods were used, k-NN and ε-NN reduced the percentages of noisy neighbors to around 22% for valence prediction and 31% for arousal prediction. The use of the graph-cut methods further reduced the percentages to around 20% for valence prediction and 29% for arousal prediction. These reductions mainly came from the selection of most similar neighbors to the unseen words but the noisy words with a higher similarity still could not be removed. The community-based method addressed this problem by considering dense connections within communities and sparse connections between communities, and thus can effectively remove the noisy neighbors, especially for those with a high degree of similarity to the unseen words. The results show that the community-based method significantly reduced the percentages of noisy neighbors to around 11% for valence prediction and 22% for arousal prediction. The performances of all VA prediction methods were improved accordingly, indicating that the community-based method can not only improve the weighted graph model but also improve the other regression- and graph-based methods

V. CONCLUSION

This study presents a community-based weighted graph model to predict VA ratings of affective words. The

community-based method selects useful neighbors for each unseen word by considering overall associations between words in the graph. These useful neighbors are then used to predict the VA ratings of unseen words using the weighted graph model with their similarities as weights. Experiments on both English and Chinese affective lexicons show that the weighted graph model yielded better performance than previously proposed methods. In addition, the use of community-based neighbor selection can further improve the performance of the weighted graph model.

The proposed method can be used for affective lexicon augmentation in the valence and arousal dimensions. Future research can benefit from such useful lexical resources to extend current VA prediction work from the word-level to sentence- and document-levels.

REFERENCES

[1] B. Pang and L. Lee, "Opinion mining and sentiment analysis," *Found. Trends Inform. Retrieval*, vol. 2, no. 1/2, pp. 1–135, 2008.

[2] R. Calvo and S. D'Mello, "Affect detection: An interdisciplinary review of models, methods, and their applications," *IEEE Trans. Affective Comput.*, vol. 1, no. 1, pp. 18–37, Jan. 2010.

[3] B. Liu, "Sentiment analysis and opinion mining," *Synthesis Lectures Human Lang. Technol.*, vol. 5, no. 1, pp. 1–167, 2012.

[4] R. Feldman, "Techniques and applications for sentiment analysis," *Commun. ACM*, vol. 56, no. 4, pp. 82–89, 2013.

[5] R. A. Calvo and S. Mac Kim, "Emotions in text: Dimensional and categorical models," *Comput. Intell.*, vol. 29, no. 3, pp. 527–543, 2013.

[6] P. Ekman, "An argument for basic emotions," *Cognition. Emotion*, vol. 6, no. 3/4, pp. 169–200, 1992.

[7] J. Li, M. Ott, C. Cardie, and E. Hovy, "Towards a general rule for identifying deceptive opinion spam," in *Proc. 52nd Annu. Meeting Assoc. Comput. Linguist.*, 2014, pp. 1566–1576.

[8] K. Schouten and F. Frasincar, "Survey on aspect-level sentiment analysis," *IEEE Trans. Knowl. Data Eng.*, vol. 28, no. 3, pp. 813–830, Mar. 2016.

[9] M. Pontiki, D. Galanis, H. Papageorgiou, S. Manandhar, and I. Androutsopoulos, "SemEval-2015 Task 12: Aspect based sentiment analysis," in *Proc. 9th Int. Workshop Semantic Eval.*, 2015, pp. 486–495.

[10] A. Mukherjee and B. Liu. "Aspect extraction through semi-supervised modeling," in *Proc. Annu. Meeting Assoc. Comput. Linguist.*, 2012, pp. 339–348.

[11] C. Banea, R. Mihalcea, and J. Wiebe, "Porting multilingual subjectivity resources across languages," *IEEE Trans. Affective Comput.*, vol. 4, no. 2, pp. 211–225, Apr. 2013.

[12] R. Xu, L. Gui, J. Xu, Q. Lu, and K. F. Wong, "Cross lingual opinion holder extraction based on multi-kernel SVMs and transfer learning," *World Wide Web*, vol. 18, no. 2, pp. 299–316, 2015.

[13] M. Al Boni1, K. Q. Zhou, H. Wang, and M. S. Gerber, "Model adaptation for personalized opinion analysis," in *Proc. 53rd Annu. Meeting Assoc. Comput. Linguist., 7th Int. Joint Conf. Natural Lang. Process.*, 2015, pp. 769–774.

[14] F. Ren and Y. Wu, "Predicting user-topic opinions in twitter with social and topical context," *IEEE Trans. Affective Comput.*, vol. 4, no. 4, pp. 412–424, Oct./Dec. 2013.

[15] L. C. Yu, C. H. Wu, and F. L. Jang, "Psychiatric document retrieval using a discourse-aware model," *Artif. Intell.*, vol. 173, no. 7, pp. 817–829, 2009.

[16] M. Qiu and J. Jiang, "A latent variable model for viewpoint discovery from threaded forum posts," in *Proc. NAACL-HLT*, 2013, pp. 1031–1040.

[17] M. Taboada, J. Brooke, M. Tofiloski, K. Voll, and M. Stede, "Lexicon-based methods for sentiment analysis," *Comput. Linguist.*, vol. 37, no. 2, pp. 267–307, 2011.

[18] F. Li, N. Liu, H. Jin, K. Zhao, Q. Yang, and X. Zhu, "Incorporating reviewer and product information for review rating prediction," in *Proc. 22nd Int. Joint Conf. Artif. Intell.*, 2011, pp. 1820–1825.

[19] L. C. Yu, J. L. Wu, P. C. Chang, and H. S. Chu, "Using a contextual entropy model to expand emotion words and their intensity for the sentiment classification of stock market news," *Knowl.-Based Syst.*, vol. 41, pp. 89–97, 2013.

[20] H. Wang and M. Ester, "A sentiment-aligned topic model for product aspect rating prediction," in *Proc. Int. Conf. Empirical Methods Natural Lang. Process.*, 2014, pp. 1192–1202.

[21] J. A. Russell, "A circumplex model of affect," *J. Personality Soc. Psychol.*, vol. 39, no. 6, pp. 1161, 1980.

[22] M. De Choudhury, S. Counts, and M. Gamon, "Not all moods are created equal! Exploring human emotional states in social media," in *Proc. Int. AAAI Conf. Web Soc. Media*, 2012, pp. 66–73.

[23] D. Preotiuc-Pietro *et al.*, "The role of personality, age and gender in tweeting about mental illnesses," in *Proc. 2nd Workshop Comput. Linguist. Clinical Psychol., Linguist. Signal Clinical Real.*, 2015, pp. 21–30.

[24] J. Ren and J. Nickerson, "Online review systems: How emotional language drives sales," in *Proc. 20th Amer. Conf. Inform. Syst.*, 2014, pp. 1–13.

[25] M. M. Bradley and P. J. Lang, "Affective norms for English words (ANEW): Instruction manual and affective ratings," *Center Res. Psychophysiology, Univ. Florida*, *Tech. Rep. C-1*, 1999.

[26] G. Paltoglou, M. Theunis, A. Kappas, and M. Thelwall, "Predicting emotional responses to long informal text," *IEEE Trans. Affective Comput.*, vol. 4, no. 1, pp. 106–115, Jan./Mar. 2013.

[27] G. Paltoglou and M. Thelwall, "Seeing stars of valence and arousal in blog posts," *IEEE Trans. Affective Comput.*, vol. 4, no. 1, pp. 116–123, Jan./Mar. 2013.

[28] N. Malandrakis, A. Potamianos, E. Iosif, and S. Narayanan, "Distributional semantic models for affective text analysis," *IEEE Trans. Audio, Speech, Lang. Process.*, vol. 21, no. 11, pp. 2379–2392, Nov. 2013.

[29] S. M. Kim, A. Valitutti, and R. A. Calvo. "Evaluation of unsupervised emotion models to textual affect recognition," in *Proc. NAACL-HLT Workshop Comput. Approaches Anal. Generation Emotion Text*, 2010, pp. 62–70.

[30] M. M. Bradley and P. J. Lang, "Measuring emotion: The self-assessment manikin and the semantic differential," *J. Behavior Therapy Experimental Psychiatry*, vol. 25, no. 1, pp. 49–59, 1994.

[31] A. B. Warriner, V. Kuperman, and M. Brysbaert, "Norms of valence, arousal, and dominance for 13,915 English lemmas," *Behavior Res. Methods*, vol. 45, no. 4, pp. 1191–1207, 2013.

[32] W. L. Wei, C. H. Wu, and J. C. Lin, "A regression approach to affective rating of Chinese words from ANEW," in *Proc. 4th Affective Comput. Intell. Interaction*, 2011, pp. 121–131.

[33] N. Malandrakis, A. Potamianos, E. Iosif, and S. Narayanan, "Kernel models for affective lexicon creation," in *Proc. 12th Annu. Conf. Int. Speech Commun. Assoc.*, 2011, pp. 2977–2980.

[34] A. Esuli and F. Sebastiani F, "Pageranking wordnet synsets: An application to opinion mining," in *Proc. 45th Annu. Meeting Assoc. Comput. Linguist.*, 2007, pp. 442–431.

[35] G. A. Miller, "WordNet: A lexical database for English," *Commun. ACM*, vol. 38, no. 11, pp. 39–41, 1995.

[36] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," in *Proc. Int. Conf. Learn. Representat.*, 2013, pp. 1–12.

[37] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean, "Distributed representations of words and phrases and their compositionality," in *Proc. Adv. Neural Inform. Process. Syst.*, 2013, pp. 3111–3119.

[38] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Empirical Methods Natural Lang. Process.*, 2014, pp. 1532–1543.

[39] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *Trans. Assoc. Comput. Linguist.*, vol. 3, pp. 211–225, 2015.

[40] R. Schwartz, R. Reichart, and A. Rappoport, "Symmetric pattern based word embeddings for improved word similarity prediction," in *Proc. CoNLL*, 2015, 258–267.

[41] S. M. Mohammad, B. J. Dorr, G. Hirst, and P. D. Turney, "Computing lexical contrast," *Comput. Linguist.*, vol. 39, no. 3, pp. 555–590, 2013.

[42] W. Dong, C. Moses, and K. Li, "Efficient k-nearest neighbor graph construction for generic similarity measures," in *Proc. 20th In. Conf. World Wide Web*, 2011, pp. 577–586.

[43] M. H. Rohban and H. R. Rabiee, "Supervised neighborhood graph construction for semi-supervised classification," *Pattern Recogn.*, vol. 45, no. 4, pp. 1363–1372, 2012.

[44] A. Blum and S. Chawla, "Learning from labeled and unlabeled data using graph mincuts," in *Proc. 18th Int. Conf. Mach. Learn.*, 2001, pp. 19–26.

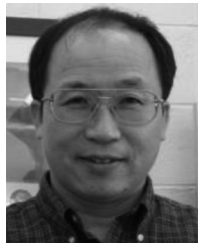
[45] B. Pang and L. Lee, "A sentimental education: Sentiment analysis using subjectivity summarization based on minimum cuts," in *Proc. Assoc. Comput. Linguist.*, 2004, pp. 271–278.

[46] L.R. Ford and D.R. Fulkerson, "Maximal flow through a network," *Can. J. Math.*, vol. 8, no. 3, pp. 399–404, 1956.

[47] X. Qian and Y. Liu, "Fast joint compression and summarization via graph cuts," in *Proc. Int. Conf. Empirical Methods Natural Lang. Process.*, 2013, pp. 1492–1502.
[48] R. F. Astudillo, S. Amir, W. Ling, B. Martins, M. Silva, and I. Trancoso, "INESC-ID: A regression model for large scale Twitter sentiment lexicon induction," in *Proc. Int. Workshop Semantic Eval.*, 2015, pp. 613–618.
[49] D. Tang, F. Wei, B. Qin, M. Zhou, and T. Liu, "Building large-scale Twitter-specific sentiment lexicon: A representation learning approach," in *Proc. 25th Int. Conf. Comput. Linguist.*, 2014, pp. 172–182.
[50] D. Rao and D. Ravichandran, "Semi-supervised polarity lexicon induction," in *Proc. 12th Conf. Eur. Chapter Assoc. Comput. Linguist.*, 2009, pp. 675–682.
[51] A. Hassan, A. Abu-Jbara, R. Jha, and D. Radev, "Identifying the semantic orientation of foreign words," in *Proc. Assoc. Comput. Linguist.*, 2011, pp. 592–597.
[52] P. D. Turney and P. Pantel, "From frequency to meaning: Vector space models of semantics," *J. Artif. Intell. Res.*, vol. 37, no. 1, pp. 141–188, 2010.
[53] O. Levy, Y. Goldberg, and I. Dagan, "Improving distributional similarity with lessons learned from word embeddings," *Trans. Assoc. Comput. Linguist.*, vol. 3, pp. 211–225, 2015.
[54] J. Turian, L. Ratinov, and Y. Bengio, "Word representations: A simple and general method for semi-supervised learning," in *Proc. Assoc. Comput. Linguist.*, 2010, pp. 384–394.
[55] A. L. Maas, R. E. Daly, P. T. Pham, D. Huang, A. Ng, and C. Potts. "Learning word vectors for sentiment analysis," in *Proc. 49th Annu. Meeting Assoc. Comput. Linguist.*, 2011, pp. 142–150.
[56] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, and P. Kuksa, "Natural language processing (almost) from scratch," *J. Mach. Learn. Res.*, vol. 12, pp. 2493–2537, 2011.
[57] M. E. Newman, "Modularity and community structure in networks," *Proc. Nat. Acad. Sci.*, vol. 103, no. 23, pp. 8577–8582, 2006.
[58] V. D. Blondel, J. L. Guillaume, R. Lambiotte, and E. Lefebvre, "Fast unfolding of communities in large networks," *J. Stat. Mech. Theory Experiment*, vol. 2008, no. 10, 2008, Art. no. 10008.
[59] C. L. Huang *et al.*, "The development of the Chinese linguistic inquiry and word count dictionary," *Chinese J. Psychol.*, vol. 54, no. 2, pp. 185–201, 2012.
[60] D. A. Schult and P. Swart, "Exploring network structure, dynamics, and function using NetworkX," in *Proc. 7th Python Sci. Conf.*, 2008, pp. 11–16.

Jin Wang is currently working toward the Ph.D. degree at the School of Information Science and Engineering, Yunnan University, China, and the Department of Computer Science and Engineering, Yuan Ze University, Taiwan. His research interests include natural language processing, text mining, and machine learning.

Liang-Chih Yu received the Ph.D. degree in computer science and information engineering from National Cheng Kung University, Taiwan. He is currently an Associate Professor in the Department of Information Management, Yuan Ze University, Taiwan. From 2007 to 2008, he was a Visiting Scholar at the Natural Language Group, Information Sciences Institute, University of Southern California (USC/ISI). He is currently a Board Member of the Association for Computational Linguistics and Chinese Language Processing, and serves as an Editorial Board Member of the International Journal of Computational Linguistics and Chinese Language Processing. His research interests include natural language processing, sentiment analysis, information retrieval, and text mining.



K. Robert Lai received the Ph.D. degree in computer science from North Carolina State University in 1992. He is currently a Professor in the Department of Computer Science and Engineering, and the Director of Innovation Center for Big Data and Digital Convergence, Yuan Ze University, Taiwan. His research interests include big data analytics, agent technologies, and mobile computing.

Xuejie Zhang received the Ph.D. degree in computer science and engineering from the Chinese University of Hong Kong, in 1998. He is currently a Professor in the School of Information Science and Engineering, and the Director of High Performance Computing Center, Yunnan University, China. His research interests include high performance computing, cloud computing, and big data analytics.