Sentiment Lexicon Expansion Based on Neural PU Learning, Double Dictionary Lookup, and Polarity Association

Yasheng Wang[1], **Yang Zhang**[1], **Bing Liu**[2]
[1]Noah's Ark Lab, Huawei Technologies
[1]{wangyasheng, zhangyang86}@huawei.com
[2]Department of Computer Science, University of Illinois at Chicago, USA
[2]liub@uic.com

Abstract

Although many sentiment lexicons in different languages exist, most are not comprehensive. In a recent sentiment analysis application, we used a large Chinese sentiment lexicon and found that it missed a large number of sentiment words used in social media. This prompted us to make a new attempt to study sentiment lexicon expansion. This paper first formulates the problem as a *PU learning* problem. It then proposes a new PU learning method suitable for the problem based on a neural network. The results are further enhanced with a new dictionary lookup technique and a novel polarity classification algorithm. Experimental results show that the proposed approach greatly outperforms baseline methods.

1 Introduction

Sentiment lexicons contain words (such as *good*, *beautiful*, *bad*, and *awful*) that convey positive or negative sentiments. They are instrumental for many sentiment analysis applications. So far many algorithms have been proposed to generate such lexicons (Liu, 2012). These algorithms are either dictionary-based or corpus-based. In the dictionary-based approach, one exploits synonym and antonym relations in the dictionary to bootstrap a given seed set of sentiment words (Hu and Liu, 2004; Kim and Hovy, 2004; Kamps et al., 2004), or learns a classifier to classify the gloss of each word in the dictionary (Esuli and Sebastiani, 2005). The corpus-based approach uses various linguistic rules or patterns (Hatzivassiloglou and McKeown, 1997; Kanayama and Nasukawa, 2006; Qiu et al., 2011; Tang et al., 2014). We will discuss these and other existing methods in the related work section.

Despite many existing studies, the problem is far from being solved. In a recent application, we used a popular Chinese sentiment lexicon for sentiment classification of Weibo posts (similar to Twitter), and found that it missed a large number of sentiment words. As the lexicon was compiled using formal text such as news, it misses a large number of sentiment words used in social media. Due to the informal nature, many "low class" words are used in social media but seldom used in formal media. New words are also created constantly. Note that new words in Chinese are easier to create from individual characters than in other languages. Thus many of these words are not in the dictionary. All these prompted us to make a new attempt to study *sentiment lexicon expansion*.

In this paper, we solve the problem in two steps: (1) identify sentiment words from a given corpus, and (2) classify their polarity. We formulate Step 1 as a *PU learning* problem (*learning from positive unlabeled examples*). To our knowledge, this is the first such formulation. This is important because it gives us a formal model to tackle the problem. PU learning is stated as follows (Liu et al., 2002): given a set P of examples of a particular class (we also use P to denote the class) and a set U of unlabeled examples which contains hidden instances from both classes P and *not-P* (called N), we want to build a classifier using P and U to classify the data in U or future test data into the two classes, i.e., P and N (or *not-P*). In our case, P is the existing sentiment lexicon, and U is a set of candidate words from a social media corpus. We identify those words in U that are also sentiment words.

A typical PU learning algorithm works by first identifying a small set of *reliable* N class examples (*RN*) from the unlabeled set U and then running a supervised learning method (e.g., SVM) it-

eratively to add more and more data to the *RN* set to finally build a classifier (Liu, 2011).

In this work, we first adapt a popular such approach to an *augmented multilayer perceptron* (AMP) method and use it to replace SVM, and show that using SVM as the learning method is inferior to using AMP. However, we can do much better. We then propose a new PU learning method, called SE-AMP (*Spy-based Elimination of P class instances using AMP*), which can better exploit the specific nature of our problem. SE-AMP goes in the opposite direction to the existing approach. It starts by treating U as the class N (*not-P*) data, and then runs the AMP learning method on P and U iteratively to gradually remove potential P class instances from U to purify U so that as iterations progress, fewer and fewer P class instances are still in U. We detail the method in Sec. 3.3. Note that SE-AMP is general and not limited to our task of sentiment lexicon expansion.

We also propose a new method based on dictionary lookup, called *Double dictionary Lookup* (DL), to enhance the results from the proposed PU learning method. The DL method is also in the framework of PU learning. Our final proposed method for Step 1 is called SE-AMP-DL.

For polarity classification (Step 2, after sentiment words are found), we propose a novel method that is based on polarity association of individual (Chinese) characters in each word.

In summary, this paper has several innovations:

1. It formulates Step 1 of sentiment lexicon expansion as a PU learning problem. To the best of our knowledge, this is the first such formulation.

2. It proposes a new neural learning method AMP and shows that AMP outperforms the traditional SVM based PU learning approach.

3. It further proposes a new and general PU learning strategy that works in the opposite direction to the popular existing approach to suit our task.

4. It also proposes a double dictionary lookup technique to improve the result further.

5. It proposes a novel polarity classification method to classify the polarity of each word.

Experimental results show that the proposed approach makes considerable improvement over existing baseline methods.

2 Related Work

There are two main approaches for sentiment lexicon generation (Liu, 2012): the *dictionary-based approach* and the *corpus-based approach*. Under the dictionary-based approach, one method is to use synonym and antonym relations and WordNet graph in the dictionary to bootstrap a set of given seed sentiment words. There are numerous variations of and enhancements to this approach (Hu and Liu, 2004; Valitutti et al., 2004; Kim and Hovy, 2004; Takamura et al., 2007; Andreevskaia and Bergler, 2006; Kaji and Kitsuregawa, 2007; Blair-Goldensohn et al., 2008; Cambria et al., 2016; Rao and Ravichandran, 2009; Perez-Rosas et al., 2012). For example, (Valitutti et al., 2004; Kim and Hovy, 2004) tried to remove error words and assign a sentiment strength to each word. Mohammad et al. (2009) exploited many antonym-generating affix patterns, Kamps et al. (2004) used a WordNet distance, and Hassan and Radev (2010) used a Markov random walk model over a word relatedness graph. Dragut et al. (2010) used a set of inference rules to determine word sentiment polarity through a deductive process, and Schneider and Dragut (2015) employed a linear programming approach. Another method is to build a supervised sentiment classifier to classify the gloss text of each word in the dictionary (Esuli and Sebastiani, 2005, 2006). Xu et al. (2010a) integrated both dictionaries and corpora to find emotion words based on label-propagation. Perez-Rosas et al. (2012) also worked on cross lingual lexicon construction.

In the corpus-based approach, one key idea is to exploit some linguistic conventions on connectives such as AND and OR Hatzivassiloglou and McKeown (1997). For example, in the sentence "This car is beautiful and spacious," if "beautiful" is known to be positive, it can be inferred that "spacious" is also positive. Kanayama and Nasukawa (2006) extended the idea to the sentence level by exploiting adversative expressions such as "but" and "however." Qiu et al. (2011) proposed a double propagation (DP) method that uses both sentiment and target relation and various connectives to extract sentiment words. (Wang and Wang, 2008) did similar works. Huang et al. (2014) detected new sentiment words using lexical patterns. Wilson et al. (2005), Ding et al. (2008), Choi and Cardie (2008) and Zhang and Liu (2011) studied contextual sentiments at the phrase or expression

level. We do not study contextual sentiments.

Another idea for the corpus-based approach is to use word co-occurrences. Turney (2002) computed the Pointwise Mutual Information (PMI) between the target word and seed words to decide its sentiment polarity. This method was extended in (Mohammad et al., 2013; Yang et al., 2014). (Hamilton et al., 2016; Xu et al., 2010b) constructed domain lexicons using lexical graphs.

Recent works also exploited neural networks and word embedding, and treated lexicon generation as a classification problem like us. Tang et al. (2014) expanded a sentiment lexicon using a softmax classifier with the proposed sentiment-specific embedding. Vo and Zhang (2016) obtained sentiment attribute scores of each word through a neural network model to predict tweets emoticons. Bravo-Marquez et al. (2015) classified words using manual features and emoticon-annotated tweets. However, all these works require different kinds of labeled data. We do not rely on emoticons or other forms of annotations.

The problem of adapting a general lexicon to a domain specific one was studied in (Choi and Cardie, 2009; Jijkoun et al., 2010; Du et al., 2010). Feng et al. (2011) also generated a connotation lexicon. These are clearly different from our work.

3 The Proposed Approach

As mentioned in Sec. 1, our problem is solved in two steps: (1) identify sentiment words and (2) classify their polarity. In the following, we first introduce a traditional PU learning method using SVM (Sec. 3.1), and the augmented multilayer perceptron (AMP) method (Sec. 3.2) to set the background for the proposed technique. The proposed PU learning algorithm is presented in Sec. 3.3 for performing the task of step 1, which uses AMP. After that, a dictionary based method called Double Dictionary Lookup (Sec. 3.4) is presented to further improve the result of the first step. The proposed polarity classification method for the second step is discussed in (Sec. 3.5).

3.1 Traditional PU Learning

PU learning has been investigated by several researchers (Liu et al., 2002; Denis et al., 2002; Li and Liu, 2003; Yu et al., 2002; Elkan and Noto, 2008; Hsieh et al., 2015). A popular approach follows a two-stage strategy: (i) identifying a set of reliable N class instances RN from the unlabeled set U; and (ii) building a classifier using P (P class) and RN (N class) and $Q = U - RN$ (unlabeled) by applying a learning algorithm (e.g., SVM) iteratively.

To understand the difference between the proposed algorithm and the above two-stage approach, we give more details to an existing algorithm (Liu, 2011). In the first stage, a *Spy* technique is used to identify the set of reliable N class instances (or examples) RN from U. It works as follows: 10% of P class instances (in our cases, they are words) is first randomly selected as a *spy* set S and put in the unlabeled set U. Then SVM is run using the set $P - S$ as the P class training data and $U \cup S$ as the N class training data. After training, the resulting classifier assigns a probability to each instance in S to decide a probability threshold th. Instances in U that has a lower probability than th are selected as RN. As suggested in (Liu, 2011), th is set to the probability that separates the last 15% instances in S. We also experimented with 10% and 20%, but the results are similar.

In the second stage, we run SVM iteratively. In each iteration, a classifier trained using P and RN is used to classify the instances in $Q = U - RN$. Those instances assigned the N class in Q are removed and added to RN. Iterations stop when no instance in Q is classified to the N class.

3.2 Augmented Multilayer Perceptron

We now present the proposed AMP (*Augmented Multilayer Perceptron*) method, which we will use to replace SVM in PU learning as AMP produces better results. AMP has three layers (Figure 1). The first layer takes the *word vector* of each word as input with an output of 50 dimensions. The second layer takes the output of the first layer as input to produce an output of 2 dimensions. Both the first and second layers use the RELU activation function. The 2 dimension output of the second layer concatenates with 5 POS features (see Sec. 4.1.3) to form a 7 dimension feature vector as input of the third layer, which is the final layer with the activation function of Sigmoid.

We note that instead of putting POS features in the first layer, we first reduce the dimension of word vector from 200 to 2 with two layers, then compose a vector with POS features as the input to the third layer. This enables POS features to play a more important role. This method gives better results than combining the word vector and POS

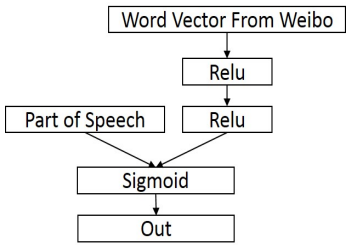

Figure 1: Aumented Multilayer Perceptron

tag features as the input in the first layer. Our experimental results also show that using AMP to replace SVM above in PU learning produces much better results.

3.3 Proposed New PU Learning Algorithm

We now present the proposed new PU learning strategy, which has only one stage that runs a supervised learning algorithm iteratively using P and U by treating U as the N class data. This strategy is more suitable for our task as we will explain in Sec. 4.1.4. The general idea is to remove words from U that are likely to belong to the P class until some stopping criterion is satisfied. The proposed algorithm is called SE-AMP (*Spy-based Elimination of P class instances using AMP*), which is given in Algorithm 1. Note that in the algorithm, we use $+1$ to denote class P (line 1) and -1 to denote class N (line 5). We don't use SVM any more as AMP performs much better. We now detail the working of the algorithm.

The algorithm still uses *Spies* and also works iteratively, but in a very different way. It first randomly sample a small proportion γ of words from set P to form the spy set S (line 2), which is added to the current U set to form U_s (line 4). The U set is updated in each iteration. An AMP classifier is trained using set P (with the class label $+1$) and set U_s (regarded as class N with the class label -1) (line 6). The resulting classifier or model M is used to score or assign a probability to each instance in U and in S (line 7).

Now we come to the crucial steps of the proposed algorithm. It tries to remove likely P class instances from U. U is essentially regarded as an noisy N class data, i.e., it has a lot of errors (which are hidden P class instances). Thus this step effectively aims to purify the N class set. In line 8, the algorithm determines a threshold θ to remove some likely P class instances from U. We will discuss the function for determining θ below.

Based on the probability threshold θ, the algorithm removes those instances in U with greater probability than θ (lines 9-13) and those instances in the spy set S (line 14-18).

The algorithm stops when the stopping criteria is met (lines 19 and 20); otherwise, it goes to the next iteration with the updated U and S. We determine the threshold δ using a validation set (Sec. 4.1.1).

Determine θ: In this new algorithm, each iteration removes instances that are likely to be of P class from the unlabeled set U. One simple way is to remove the top $k\%$ of U based on the classifier result of each iteration. However, this method is undesirable because we cannot control the probabilities of the eliminated instances. We propose to use Gaussian fitting to determine the threshold θ.

After each iteration, every spy word $w_i \in S$ is assigned a probability x_i ($= Pr(P|w_i)$) to be in class P by the classifier M, a Gaussian fitting is done on these probabilities. Parameters of Gaussian distribution are obtained using Maximum Likelihood Estimate (MLE) as follows: $\mu = \frac{1}{n}\sum_{i=1}^{n} x_i$, and $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}(x_i - \mu)^2$. We set the threshold $\theta = \mu + \sigma$. Thus those words have probabilities higher than θ are considered very likely to belong to the P class. This threshold is very conservative and only allows those very likely P class instances to be removed from U.

We will discus why the proposed SE-AMP is better than S-AMP in Sec. 4.1.4 after we see the experiment results. Note that S-AMP uses the traditional PU-learning strategy discussed in Sec. 3.1 but it replaces SVM with AMP.

3.4 Double Dictionary Lookup

To improve PU learning of SE-AMP further, we propose to employ a dictionary. Using a dictionary is natural because human beings always use dictionaries to understand a word. The proposed *Double Lookup* (DL) technique is given in Algorithm 2. Why *double lookup* is needed will be clear shortly.

The algorithm works as follows. Let the set of given sentiment lexicon be P, the given dictionary be D, and the set of candidate words be U (in this case, it is the test set). For each candidate word $w \in U$, we first look w up in the dictionary D (lines 1-2). If w can be found in D (line 2), we use a lexicon-based sentiment classifier (C) (see below) to classify the gloss or explanation note (G_w) of w (lines 3-4). The function *classify* returns a set of sentiment words O_1 from G_w (line 4), which are also in P. If O_1 is not empty, it means that G_w

Algorithm 1 SE-AMP(P, U)

1: Every instance in P is assigned the class label $+1$
2: $S = Sample(P, \gamma)$; // $\gamma = 0.1$ in this experiment; instances in S are spies
3: **loop**
4: $U_s = U \cup S$
5: Every instance u in U_s is assigned the class label -1 // -1 denotes the N class
6: M = AMP(P, U_s) // Build a binary AMP classifier M
7: score(M, U, S) // Score each instance *i* in U and S using M to give each *i* a probability score
8: θ = DetermineThreshold(S) // decide a probability threshold θ using S;
9: **for** each instance $u \in U$ **do**
10: **if** its probability $Pr(+1|u) > \theta$ **then**
11: $U = U - \{u\}$
12: **end if**
13: **end for**
14: **for** each instance $s \in S$ **do**
15: **if** its probability $Pr(+1|s) > \theta$ **then**
16: $S = S - \{s\}$
17: **end if**
18: **end for**
19: **if** $|S| \leq \delta(\gamma|P|)$ **then** // Stopping criterion; $\gamma|P|$ is actually the original size of S in line 2
20: exit-loop
21: **end if**
22: **end loop**

is classified to the sentiment class. If it is empty, it is classified to the non-sentiment class.

This one dictionary lookup is not safe to determine whether word w is a sentiment word or not because some sentiment words in the lexicon P are noisy and don't have clear sentiments. This gives us too low precision. That is why we perform the second dictionary lookup, which is on the words in O_1 to ensure that at least one word in O_1 is very likely to be a true sentiment word because a noisy sentiment word in P is unlikely to be explained by another word in P. But a true sentiment word in P is very likely to be explained by another sentiment word in P.

Line 7 looks up each word $o \in O_1$ in D and finds its gloss or explanation note G_o. G_o is then classified in line 8, which returns a list of sentiment words O_2 from G_o, also in P. If O_2 is not empty, meaning that G_o is a sentiment sentence (line 9), we return w as a sentiment word (line 10).

Lexicon-based sentiment classifier (C): C is a binary classifier with two classes *sentiment* or *non-sentiment*. Given a sentence s (e.g., the explanation note of a word in the dictionary), it simply finds sentiment words in s that are also in the given sentiment lexicon P. If some sentiment words are found, it returns them in a set indicating the sentence s is a sentiment sentence. Although simple, this works quite well because the explanation note of a word in a dictionary is usually quite simple.

Integrating SE-AMP and DL: Our final proposed method (SE-AMP-DL) combines SE-AMP and DL. As we will see that DL has high precision but low recall because most of the new words cannot be found in the dictionary, we use the results of DL to correct the results of the SE-AMP algorithm. Words that are classified as belong to the N class by SE-AMP are moved to the P class if the DL method believes them to be sentiment words.

3.5 Polarity Classification

After sentiment words are discovered, this step determines the polarity (positive or negative) of each discovered sentiment word. We propose a new classification method exploiting the fact that new Chinese words are created with 2 or more Chinese characters. The meaning of a Chinese word is closely related to the meaning of each individual character of the word. So the polarity of a Chinese word is strongly related to the polarity of the characters that form the word, which is the motivation of the new classification method. We use the polarity association of the characters in each word to predict the polarity of the word based on super-

Algorithm 2 DL(P, D, U, C)

for each candidate word $w \in U$ **do**
  if w can be found in D **then** // First dictionary lookup; D is the dictionary
    Let G_w be the gloss or explanation of word w in D
    $O_1 =$ classify(G_w, C) // O_1 is the set of sentiment words in G_w and $O_1 \subseteq P$ (lexicon)
    if $O_1 \neq \emptyset$ **then** // G_w is classified to the sentiment class
      for each word $o \in O_1 (\subseteq P)$ **do**
        Let G_o be the gloss or explanation of word o in D // Second dictionary lookup
        $O_2 =$ classify(G_o, C) // $O_2 \subseteq P$ (lexicon) and C is the sentiment classifier
        if $O_2 \neq \emptyset$ **then** // G_o is classified as a sentiment sentence
          w is a new sentiment word
        end if
      end for
    end if
  end if
end for

vised learning. We call the method CPA (*Character Polarity Association*). This method is very useful for languages whose words are constructed by characters such as Chinese and Japanese.

Feature Vector: For a character, we calculate the percentages of it appearing in the positive words and negative words in the existing lexicon P to form a 2-dimensional polarity vector. For example, the polarity vector (0.9, 0.1) means that 90% of the words in the existing lexicon that contains the character are positive and the other 10% are negative. Thus, for a word with 2 characters, which covers most cases in Chinese, a 4-dimensional vector is formed. For those words with more than 2 characters, we choose 2 characters with the strongest polarity to form a 4-dimensional vector.

Classifier Building: Using all positive and negative words in the existing lexicon P as the training sample, each word represented as a vector of four features, we apply a Naive Bayesian Classifier to build a polarity classifier. For testing, a word is represented in the same way. If a test word contains charters that don't exist in the lexicon, we give each character (0.5, 0.5) as the polarity vector.

4 Experimental Evaluation

We now evaluate the proposed technique SEAMP-DL to expand an existing Chinese sentiment lexicon, DUTIR (Dalian University of Technology, Information Retrieve Lab) Affective Lexicon Ontology (Xu et al., 2008). DUTIR lexicon is perhaps the largest Chinese sentiment lexicon with 27466 words. Although large, since it was built based on formal text such as news, essays, and novels, it does not contain many sentiment words often used in social media as we will see later. We will also see that many new sentiment words that we discovered are not even in an authoritative Chinese language dictionary. Thus, compiling a comprehensive sentiment lexicon is needed. Below, we first evaluate sentiment words discovery (Step 1) and then polarity classification (Step 2).

4.1 Sentiment Words Discovery

4.1.1 Data and Parameter Settings

We use a large Chinese Weibo corpus (Chinese version of Twitter) from (Wang et al., 2013) for our lexicon expansion, which has about 4.4 million pairs of post and response messages. Although it was originally used to study natural language conversations, it is quite suitable for our purpose as online conversations are sentiment rich.

Word embedding: We first used the Stanford Chinese word segmenter to split sentences into sequences of words (the POS-tag of each word is also obtained in the process). For word embedding, we used word2vec (Mikolov et al., 2013). Each word vector has 200 dimensions.

P set, U set, validation set, and test set: We randomly sampled 200K messages, and used all 54303 words contained in them as our experiment dataset. Out of the 54303 words (stopwords have been removed), 4957 of them also appear in the DUTIR lexicon and are thus sentiment words. The remaining 49346 words are unlabeled.

Validation set: The validation set consists of randomly selected 300 words from the 4957 sen-

timent words and 700 words from the 46742 unlabeled words. Although the 700 words are unlabeled, they are treated as N class words.

Test set: 1000 words are randomly sampled from 48646 ($= 49346 - 700$) unlabeled words as the test set, which is labeled manually. Two native speakers labeled the 1000 test words. The Kappa agreement score is 0.695. For any word with disagreement, the annotators discussed to come to a final decision. The annotated test set has 199 sentiment words, 78 positive and 121 negative words.

P set and U set: The remaining 4657 ($= 4957 - 300$ sentiment words serve as the P set and the remaining 45042 ($= 46742 - 700 - 1000$) words serve as the U set for learning.

Parameter settings: As indicated earlier, 10% (γ) of the P class examples are randomly selected as spies S (465). The probability threshold θ is set using the Gaussian fitting (Sec. 3.3). The iteration stopping criterion of SE-AMP (Sec. 3.3) is decided using the validation set ($\delta = 30\%$).

Evaluation measures: We use the classic $precision$, $recall$, and F score for evaluation.

4.1.2 Compared Systems

We compare the following seven (7) techniques:

DP: The Double Propagation (DP) Method in (Qiu et al., 2011). This method uses dependency patterns for extraction.

PMI: The classic PMI method (Turney, 2002) using the full Weibo corpus and 100 positive and 100 negative words in the DUTIR sentiment lexicon as the reference words. These words appear most frequently in the Weibo corpus. In (Turney, 2002), only 1 positive and 1 negative reference words are used. We also tried to use 1, 50, 150, 200, and all words in the positive and negative classes as reference words, respectively. However, they give poorer results. Since the PMI method can only determine the polarity, but cannot decide whether a test word is a sentiment word or not. We make that decision by using the mean score the PMI method of all positive words in the lexicon as the positive threshold and the mean score of all negative words as the negative threshold.

S-SVM: The traditional PU learning method described in Sec. 3.1 using SVM.

S-AMP: Same as S-SVM, but SVM is replaced with AMP.

SE-AMP: The proposed PU learning method without DL.

DL: Only the double dictionary lookup method, using the Contemporary Chinese Dictionary.

SE-AMP-DL: This is our final proposed method, which combines SE-AMP and DL.

We could not compare with recent approaches Tang et al. (2014); Vo and Zhang (2016); Bravo-Marquez et al. (2015) as they all require some supervised information on the data. Our method is unsupervised except the use of the lexicon. These methods were also evaluated indirectly based on the social media post sentiment or emotion classification results. None reported precision, recall, or F score of the discovered sentiment words.

We do not compare with a dictionary-based approach because most of the test words are not even in the dictionary. Only 87 of 199 sentiment words in the test set can be found in the Contemporary Chinese dictionary. Note that in Chinese, one can form a word using characters fairly easily.

4.1.3 Features

For both SVM and AMP, the feature vector for each word is the word vector and POS tags. POS tags are divided into 5 classes (noun, verb, adjective, adverb, others) and form a 5 dimension binary vector (e.g. $[1, 0, 0, 0, 0]$ for noun). In the SVM approach, POS tags are concatenated to the word embedding features to form a 205 dimension feature vector (5 POS tags and 200 word embedding features). We used the RBF kernel, which gives the best result as compared to other kernels.

4.1.4 Experiment Results

We now present and discuss the results. The syntactic pattern based DP approach performed very poorly because social media posts are brief and the use of patterns to link sentiment words are quite infrequent. Thus, we will not include its results. Below, we first compare PMI, S-SVM and S-AMP, and then the results of the proposed PU learning method SE-AMP (without using DL). The results of incorporating DL are discussed last.

Existing Approach - PMI vs. S-SVM vs. S-AMP: S-SVM and S-AMP use the traditional PU learning approach described in Sec. 3.1 for model building. 10% of P class examples are sampled as spies to produce the RN set.

Table 1 shows the results of S-SVM and S-AMP iterations. We observe that the AMP version performs much better than the SVM version. The first three iterations improve the results. But after that, the results deteriorate for both systems.

Thus the table only shows the first few iterations since the results keep deteriorating after iteration 2. The best results are obtained by S-AMP, which is 0.548 in F score. S-SVM's best F score is only 0.509, which is poorer. We also see that the precision and recall of S-AMP are both consistently better than those of S-SVM.

We note that these two algorithms cannot catch the best results when they stop following the algorithm in (Liu, 2011). We did not use the validation set here to find the best stopping criterion because even their best results are poorer than those of SE-AMP. PMI does similarly to S-AMP.

Proposed Approach - SE-AMP: Table 2 shows the results of the proposed PU learning approach (SE-AMP). As noted above, the iteration stopping criterion δ is determined using the validation set. Iteration 0 means the classifier uses all unlabeled examples in U as the N set. Compared with iteration 0 of the traditional strategy (S-AMP), the F score of SE-AMP improves slightly (from 52.0% to 54.8%), but both are low. This is because the reliable N set RN for the traditional approach is too small (not representative of all N class examples) while for the proposed approach, the N set has too many hidden P class instances. The traditional PU learning tries to include more and more likely N examples iteratively to move to the P direction while the proposed approach doing the opposite, eliminating likely P instances from the unlabeled set U. As the table shows, the results of SE-AMP gets better and better after each iteration (it stops when the stopping condition is met). Precision, recall and F score all improve consistently, which result in improvements of 12.0%, 8.5% and 9.6% respectively. Compared with the best result of S-AMP, the precision of SE-AMP improves from 55.4% to 67.4%, recall improves from 51.8% to 59.3% and the F score improves from 53.5% to 63.1%.

We now explain why SE-AMP is better than S-AMP. We believe that the main reason is the high level of noise in P, i.e., many words in P don't have clear sentiments. The traditional PU learning (S-AMP) tries to add classified N class instances into the RN set in each iteration. This works fine for the first few iterations but then goes wrong because the noise in P resulted in a lot of hidden P instances added into the RN set. Then the results deteriorate as more and more wrong instances are added as iterations progress. In contrast, the proposed SE-AMP removes likely P instances (including those noisy ones) from the U set to obtain a purer and purer N set. Due to the very conservative setting of the θ parameter (see Sec. 3.3), the number of words removed from U in each iteration is small, so is the number of spy words removed from S as we can see in Table 2. Thus, U becomes purer and purer slowly, and the validation set helps find a good stopping iteration.

	Iteration	Precision	Recall	F-score
S-SVM	0	46.1	41.7	43.8
	1	49.7	45.7	47.6
	2	52.7	49.2	50.9
	3	48.9	46.2	47.5
S-AMP	0	52.8	51.3	52.0
	1	54.4	52.8	53.6
	2	56.4	53.3	54.8
	3	54.2	52.3	53.2
PMI		56.7	50.8	53.6

Table 1: Results of PMI and the traditional PU learning approach: S-SVM and S-AMP.

Iteration	Precision	Recall	F-score	Spy Words
0	55.4	51.8	53.5	456
1	57.1	52.8	54.8	372
2	59.8	53.8	56.6	268
...	...	...	...	...
6	67.4	59.3	63.1	143

Table 2: Results of the proposed SE-AMP.

Incorporating Double Dictionary Lookup (DL) - SE-AMP-DL: The double dictionary lookup (DL) method improves the results further. DL uses the most authoritative Chinese dictionary: The Contemporary Chinese Dictionary. However, only 379 out of the 1000 test words are in the dictionary, among which 87 are sentiment words. As Table 3 shows, the DL method alone has a high precision but low recall as a lot of sentiment words are not in the dictionary. After correction of the results from SE-AMP by DL, the F score improves from 63.1 of SE-AMP to 65.6 of SE-AMP-DL.

Note: We also tried to clean up the lexicon first using DL to reduce the noise in the P set

	Precision	Recall	F-score
DL	74.1	20.1	31.6
SE-AMP	67.4	59.3	63.1
SE-AMP-DL	69.3	62.3	65.6

Table 3: Results with double dictionary lookup.

			Prec.	Rec.	F-score
PMI	199 set	pos.	60.7	65.4	63.0
		neg.	76.5	72.7	74.6
	identified set	pos.	43.8	35.9	39.2
		neg.	45.1	42.1	43.6
CPA	199 set	pos.	83.8	79.5	81.6
		neg.	87.2	90.1	88.6
	identified set	pos.	60.9	50.0	54.9
		neg.	65.2	60.3	62.6

Table 4: PMI and CPA classification results.

before performing the proposed algorithms. But after cleaning, only 1968 sentiment words out of 4957 remained. We inspected the result and found that the cleaning removed a lot of true sentiment words, making the P set too small for our algorithms and thus produced much poorer results.

4.2 Polarity Classification

Here we use the well-known PMI method in (Turney, 2002) as the baseline, which was designed for polarity classification. Again, for PMI computation, we use 100 positive and 100 negative words in our lexicon that appear most frequently in the Weibo corpus as the reference words, and compute the PMI scores between the candidate words and the references. Using many other numbers of sentiment words in the lexicon as the reference words give poorer results (see also Sec. 4.1.2). A word is considered as a positive sentiment word if its score is positive, or negative if the score is negative.

We apply the proposed CPA method and PMI to all 199 true sentiment words in the test set (199 set), and the sentiment words identified by SEAMP-DL (identified set), which has many errors, i.e., non-sentiment words, Experimental results in Table 4 show that CPA outperforms PMI greatly in both cases. Those non-sentiment words are considered wrong in the "identified set" case.

5 Conclusion

This paper made a new attempt to study sentiment lexicon expansion based on a social media corpus. It first showed that the problem can be formulated as PU learning. It then proposed an *augmented multilayer perceptron* method to give PU learning an neural network solution. It then proposed a new PU learning method, which outperforms a classic existing PU learning method. To improve the results further, it proposed a double dictionary lookup technique. Additionally, a novel polarity classification method was also designed. Experimental results demonstrated the effectiveness of these proposed methods.

Acknowledgments

Bing Liu's work was supported in part by National Science Foundation (NSF) under grant no. IIS-1407927 and IIS-1650900. We would like to show gratitude to Dr. Feng Liu, Dr. Jiansheng Wei and Youliang Yan from Noah's Ark Lab, Huawei Technologies who provided excellent advices that greatly assisted the research.

References

Alina Andreevskaia and Sabine Bergler. 2006. Mining wordnet for a fuzzy sentiment: Sentiment tag extraction from wordnet glosses. In *EACL*. volume 6, pages 209–216.

Sasha Blair-Goldensohn, Kerry Hannan, Ryan McDonald, Tyler Neylon, George A Reis, and Jeff Reynar. 2008. Building a sentiment summarizer for local service reviews. In *WWW workshop on NLP in the information explosion era*. volume 14, pages 339–348.

Felipe Bravo-Marquez, Eibe Frank, and Bernhard Pfahringer. 2015. Positive, negative, or neutral: Learning an expanded opinion lexicon from emoticon-annotated tweets. In *IJCAI 2015*. AAAI Press, volume 2015, pages 1229–1235.

Erik Cambria, Soujanya Poria, Rajiv Bajpai, and Björn Schuller. 2016. Senticnet 4: A semantic resource for sentiment analysis based on conceptual primitives. In *the 26th International Conference on Computational Linguistics (COLING), Osaka*.

Yejin Choi and Claire Cardie. 2008. Learning with compositional semantics as structural inference for subsentential sentiment analysis. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 793–801.

Yejin Choi and Claire Cardie. 2009. Adapting a polarity lexicon using integer linear programming for domain-specific sentiment classification. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2*. Association for Computational Linguistics, pages 590–598.

François Denis, Remi Gilleron, and Marc Tommasi. 2002. Text classification from positive and unlabeled examples. In *Proceedings of IPMU*.

Xiaowen Ding, Bing Liu, and Philip S Yu. 2008. A holistic lexicon-based approach to opinion mining.

In *Proceedings of the 2008 international conference on web search and data mining*. pages 231–240.

Eduard C Dragut, Clement Yu, Prasad Sistla, and Weiyi Meng. 2010. Construction of a sentimental word dictionary. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. pages 1761–1764.

Weifu Du, Songbo Tan, Xueqi Cheng, and Xiaochun Yun. 2010. Adapting information bottleneck method for automatic construction of domain-oriented sentiment lexicon. In *Proceedings of the third ACM international conference on Web search and data mining*. pages 111–120.

Charles Elkan and Keith Noto. 2008. Learning classifiers from only positive and unlabeled data. In *Proceedings of the 14th ACM SIGKDD international conference on Knowledge discovery and data mining*. pages 213–220.

Andrea Esuli and Fabrizio Sebastiani. 2005. Determining the semantic orientation of terms through gloss classification. In *Proceedings of the 14th ACM international conference on Information and knowledge management*. pages 617–624.

Andrea Esuli and Fabrizio Sebastiani. 2006. Sentiwordnet: A publicly available lexical resource for opinion mining. In *In Proceedings of the 5th Conference on Language Resources and Evaluation (LREC06*. pages 417–422.

Song Feng, Ritwik Bose, and Yejin Choi. 2011. Learning general connotation of words using graph-based algorithms. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, pages 1092–1103.

William L Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. 2016. Inducing domain-specific sentiment lexicons from unlabeled corpora. In *Empirical Methods in Natural Language Processing (EMNLP)*.

Ahmed Hassan and Dragomir Radev. 2010. Identifying text polarity using random walks. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 395–403.

Vasileios Hatzivassiloglou and Kathleen R McKeown. 1997. Predicting the semantic orientation of adjectives. In *Proceedings of the eighth conference on European chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 174–181.

Cho-Jui Hsieh, Nagarajan Natarajan, and Inderjit S Dhillon. 2015. Pu learning for matrix completion. In *ICML*. pages 2445–2453.

Minqing Hu and Bing Liu. 2004. Mining and summarizing customer reviews. In *Proceedings of the tenth ACM SIGKDD international conference on Knowledge discovery and data mining*. pages 168–177.

Minlie Huang, Borui Ye, Yichen Wang, Haiqiang Chen, Junjun Cheng, and Xiaoyan Zhu. 2014. New word detection for sentiment analysis. In *ACL (1)*. pages 531–541.

Valentin Jijkoun, Maarten de Rijke, and Wouter Weerkamp. 2010. Generating focused topic-specific sentiment lexicons. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 585–594.

Nobuhiro Kaji and Masaru Kitsuregawa. 2007. Building lexicon for sentiment analysis from massive collection of html documents. In *EMNLP-CoNLL*. pages 1075–1083.

Jaap Kamps, Maarten Marx, Robert J Mokken, Maarten De Rijke, et al. 2004. Using wordnet to measure semantic orientations of adjectives. In *LREC*. Citeseer, volume 4, pages 1115–1118.

Hiroshi Kanayama and Tetsuya Nasukawa. 2006. Fully automatic lexicon expansion for domain-oriented sentiment analysis. In *Proceedings of the 2006 conference on empirical methods in natural language processing*. Association for Computational Linguistics, pages 355–363.

Soo-Min Kim and Eduard Hovy. 2004. Determining the sentiment of opinions. In *Proceedings of the 20th international conference on Computational Linguistics*. Association for Computational Linguistics, page 1367.

Xiaoli Li and Bing Liu. 2003. Learning to classify texts using positive and unlabeled data. In *IJCAI*. volume 3, pages 587–592.

Bing Liu. 2011. *Web data mining: exploring hyperlinks, contents, and usage data. Second Edition*. Springer.

Bing Liu. 2012. *Sentiment analysis and data mining*. Morgan Claypool Publishers.

Bing Liu, Wee Sun Lee, Philip S. Yu, and Xiaoli Li. 2002. Partially supervised classification of text documents. In *Proceedings of the Nineteenth International Conference on Mach ine Learning (ICML-2002)*.

Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. pages 3111–3119.

Saif Mohammad, Cody Dunne, and Bonnie Dorr. 2009. Generating high-coverage semantic orientation lexicons from overtly marked words and a thesaurus. In

Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing: Volume 2-Volume 2. Association for Computational Linguistics, pages 599–608.

Saif M Mohammad, Svetlana Kiritchenko, and Xiaodan Zhu. 2013. Nrc-canada: Building the state-of-the-art in sentiment analysis of tweets. *arXiv preprint arXiv:1308.6242* .

Veronica Perez-Rosas, Carmen Banea, and Rada Mihalcea. 2012. Learning sentiment lexicons in spanish. In *LREC*. volume 12, page 73.

Guang Qiu, Bing Liu, Jiajun Bu, and Chun Chen. 2011. Opinion word expansion and target extraction through double propagation. *Computational linguistics* 37(1):9–27.

Delip Rao and Deepak Ravichandran. 2009. Semi-supervised polarity lexicon induction. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*. Association for Computational Linguistics, pages 675–682.

Andrew Schneider and Eduard C Dragut. 2015. Towards debugging sentiment lexicons. In *ACL (1)*. pages 1024–1034.

Hiroya Takamura, Takashi Inui, and Manabu Okumura. 2007. Extracting semantic orientations of phrases from dictionary. In *HLT-NAACL*. volume 2007, pages 292–299.

Duyu Tang, Furu Wei, Bing Qin, Ming Zhou, and Ting Liu. 2014. Building large-scale twitter-specific sentiment lexicon: A representation learning approach. In *COLING*. pages 172–182.

Peter D Turney. 2002. Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews. In *Proceedings of the 40th annual meeting on association for computational linguistics*. Association for Computational Linguistics, pages 417–424.

Alessandro Valitutti, Carlo Strapparava, and Oliviero Stock. 2004. Developing affective lexical resources. *PsychNology Journal* 2(1):61–83.

Duy Tin Vo and Yue Zhang. 2016. Dont count, predict! an automatic approach to learning sentiment lexicons for short text. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics*. volume 2, pages 219–224.

Bo Wang and Houfeng Wang. 2008. Bootstrapping both product features and opinion words from chinese customer reviews with cross-inducing. In *IJCNLP*. volume 8, pages 289–295.

Hao Wang, Zhengdong Lu, Hang Li, and Enhong Chen. 2013. A dataset for research on short-text conversations. In *EMNLP*. pages 935–945.

Theresa Wilson, Janyce Wiebe, and Paul Hoffmann. 2005. Recognizing contextual polarity in phrase-level sentiment analysis. In *Proceedings of the conference on human language technology and empirical methods in natural language processing*. Association for Computational Linguistics, pages 347–354.

Ge Xu, Xinfan Meng, and Houfeng Wang. 2010a. Build chinese emotion lexicons using a graph-based algorithm and multiple resources. In *Proceedings of the 23rd International Conference on Computational Linguistics*. Association for Computational Linguistics, pages 1209–1217.

Hongzhi Xu, Kai Zhao, Likun Qiu, and Changjian Hu. 2010b. Expanding chinese sentiment dictionaries from large scale unlabeled corpus. In *PACLIC*. pages 301–310.

Linhong Xu, Hongfei Lin, Yu Pan, Hui Ren, and Jianmei Chen. 2008. Constructing the affective lexicon ontology. *Journal of the China Society for Scientific and Technical Information* 2:180–185.

Min Yang, Baolin Peng, Zheng Chen, Dingju Zhu, and Kam-Pui Chow. 2014. A topic model for building fine-grained domain-specific emotion lexicon. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers)*. Association for Computational Linguistics (ACL).

Hwanjo Yu, Jiawei Han, and Kevin Chen-Chuan Chang. 2002. Pebl: positive example based learning for web page classification using svm. In *Proceedings of the eighth ACM SIGKDD international conference on Knowledge discovery and data mining*. pages 239–248.

Lei Zhang and Bing Liu. 2011. Identifying noun product features that imply opinions. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2*. Association for Computational Linguistics, pages 575–580.