Disentangling Aspect and Opinion Words in Sentiment Analysis using Lifelong PU Learning

Shuai Wang[1], Mianwei Zhou[2], Sahisnu Mazumder[1], Bing Liu[1], Yi Chang[3]

[1] Department of Computer Science, University of Illinois at Chicago, USA
[2] Yahoo! Research, USA
[3] Artificial Intelligence School, Jilin University, China
shuaiwanghk@gmail.com, mianwei@yahoo-inc.com,
sahisnumazumder@gmail.com, liub@cs.uic.edu, yichang@acm.org

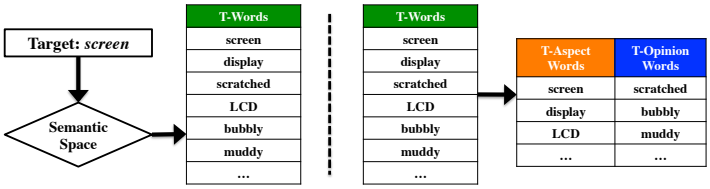
Abstract. While sentiment analysis can mine valuable information from online reviews, performing a fine-grained sentiment analysis task is very challenging due to the complex patterns in text. In this work, we focus on a Fine-grained Target-based Sentiment Analysis (FTSA) task, which is to identify target-specific aspect words and opinion words. This task is useful in practice. However, existing solutions cannot generate satisfactory results, especially when we have limited or no labeled data. To provide a holistic solution, we design a novel two-stage approach. Stage one groups the target-related words (call t-words) for a given target, which is relatively easy. Stage two separates the aspect and opinion words from the grouped t-words, which is more challenging due to the lack of sufficient word-level aspect and opinion labels. To address it, we formulate the task in a Positive-Unlabeled (PU) learning setting and incorporate the idea of lifelong learning, which achieves promising results.

1 Introduction

Online reviews have become an important type of big data and the sentiment analysis on them plays a crucial role for both customers and manufacturers. Although various sentiment analysis tasks have been studied [9], performing a fine-grained sentiment analysis task is still very challenging because of the complex patterns existing in text, such as sentiment composition, semantic understanding, and word-sense disambiguation. That is probably also why sentiment analysis is still an active research field. In this work, we focus on an important Fine-grained Target-based Sentiment Analysis (FTSA) task.

The FTSA task is defined as: *given a target name, to identify its aspect words and opinion words in a given domain corpus*. Here a target name (or simply called *target*) can be understood as an aspect category[1], such as *screen*, *voice*, or *weight*. For example, a customer or manufacturer could be interested in opinions about the target "screen" of a camera (camera is a product or called a **domain**), and wants to find out all related aspect words and opinion words mentioned in

[1] The terms *target*, *target name*, and *aspect category* will be used interchangeably.



(a) Grouping (b) Disentangling

Fig. 1: Two-stage approach to Fine-grained Target-based SA (FTSA)

the customer reviews. Ideally, one may find aspect words like "LCD," "display", and "resolution," and opinion words like "scratched," "blurry", and "bubbly."

Notice the FTSA problem is very challenging in practice, especially when there is limited or no labeled data. It is also somewhat unrealistic to manually annotate all possible aspect and opinion words for all possible targets in every domain, not to mention that there are always new domains/products coming to the real world. Designing an unsupervised or semi-supervised method for this task is thus more practical and useful. To this end, we developed a (semi-supervised) two-stage approach which does not require manual labeling.

Stage one is defined as the process of target-related words grouping. That is, given a target name, we first identify all target-related words (called *t-words*). Note that a *t-word* can be either an aspect word or opinion word. For instance, when the target is *screen*, the t-words "display", "LCD", "scratched", and "bubbly" could be extracted (shown in Figure 1a). We can achieve this goal by learning the semantic representation of words [7, 2]. Specifically, given a target, we extract its semantically similar words from its leaned representation as t-words.

One key issue is that, most semantics learning techniques will inevitably couple the target-related aspect words (called *t-aspect words*) and opinion words (called *t-opinion words*). Here we interpret its cause from a linguistic perspective. Most semantics learning models are developed based on the idea of *distributional hypothesis*: linguistic items occurring with similar contexts have similar meanings [5], so they in fact group two different types of semantic similarity together, namely, conceptual and associative similarity. Conceptual similarity means two words are conceptually similar (likely replaceable), like "dog" and "canine". Associative similarity means two words tend to appear in similar contexts, like "dog" and "bark." The distinction between them is well-known in cognitive science [16] and recently discussed in NLP [7]. In regard to sentiment analysis, we can see that t-aspect words "display" and "LCD" and t-opinion words "scratched" and "bubbly" are all mixed based on our given example (Figure 1a).

In spite of the discussed drawbacks, we argue that the semantics-based models are still quite suitable and helpful for the FTSA task, with the reason being three-fold. First, the mixture benefits the t-words grouping, where both two types of semantic correlation can be jointly extracted. To be concrete, aspect words like "display" could be found because of the conceptual similarity (similar to "screen") and opinion words like "scratched" could also be discovered due

to associative similarity (associated with "screen"). Second, many existing or new target extraction techniques can be utilized [9], which paves the way for more accurate results for FTSA (detailed in Section 3). Third, those semantics-based models are usually learned in an unsupervised or semi-supervised manner, which meets our learning requirement. However, when we take advantage of those semantics-based models for stage one, we have to overcome their aforementioned drawbacks, which leads to the proposed stage two.

Stage two is defined as: *Given a list of target-related words (t-words), separating them into target-related aspect words (t-target words) and target-specific opinion words (t-opinion words).* Figure 1b shows an example. Notice that the list of t-words is assumed to be given, which is grouped by an existing semantics-based learning technique, so we refer to this problem as ***disentangling aspect and opinion words from extracted/grouped target-related words***.

An intuitive solution to this problem is to model it as a word-level binary classification task. That is, to build a classifier to learn and predict t-aspect and t-opinion words. However, this is difficult in practice, because this means that we need both aspect and opinion word-level labels for every domain, which requires intensive human efforts. Noticing this, we instead formulate the classification problem in a Positive-Unlabeled (PU) learning setting. The idea is to use general/common opinion words (treating them as positive examples) to distill other opinion words from unlabeled words. However, a notable issue in this PU setting is that the errors from false positive (FP) examples (wrongly predicted opinion words) can be propagated, resulting in more errors and degenerating its performance. To address this issue, we exploit the idea of *lifelong machine learning* [3] and incorporate it into the PU learning process. We name it as Lifelong PU learning (LPU). It works by accumulating the knowledge learned from multiple domains, and uses it to restrict the propagation of FP examples and to ensure the reliability of the newly learned opinion words.

The main contributions of this paper are summarized as: (1) It proposes to perform the FTSA task in a two-stage manner, which does not require manual labeling. (2) It proposes a Lifelong PU (LPU) learning approach to solving the problem of disentangling target-specific aspect and opinion words . To the best of our knowledge, none of the existing studies has employed the LPU technique. (3) Experimental results conducted on multiple real-world review datasets with two target extraction techniques show its effectiveness and extensibility.

2 Related Work

Fine-grained and target-based sentiment analysis. Various types of sentiment analysis (SA) research exist in the literature [9]. Unlike the coarse-grained SA such as classifying a review document as overall positive or negative [20], fine-grained SA consists of various components such as aspect identification, opinion identification, and polarity classification [18]. In terms of targeted-based SA, most of the existing studies [17, 19] focused on the target-based polarity classification. However, our work does not lie in this direction. As discussed in

Section 1, we aim to provide a generic solution for disentangling target-specific aspect and opinion words. In fact, our work can be integrated into other related target-based analysis models and we will show it in our experiments.

Semantic space and representation. Semantics-based learning models project words to a semantic space and represent each word as a dense vector. Such semantics-bearing vectors can be created by matrix factorization (e.g. LSI) [4] and topic modeling (e.g., LDA) [2]. Recently, neural word embeddings [13] emerge to learn better semantic representation for words.

Lifelong machine learning. Our work is related to lifelong learning (LL) [3]. Regarding sentiment analysis, several LL models have been proposed for improving topic quality [18] and polarity classification [19], but they are not for the FTSA task and not applicable to the word disentangling problem. We also incorporate LL into the PU learning process, which is the first attempt. Although related, our work differs from the research field of transfer learning (TL) [12]. The distinction between LL and TL can be found in [3].

3 Stage one: Grouping

As discussed in Section 1, we group the target-related words (t-words) for a specified target in this stage. The basic idea is to extract its semantically correlated words towards the target in a learned semantic space. Specifically, we use the neural word embedding model [13] to learn word vectors for a given domain corpus, resulting in an embedding matrix $E \in \mathbb{R}^{v \times d}$ where v and d are the size of vocabulary and embedding dimension. Then a semantic similarity matrix $M \in \mathbb{R}^{v \times v}$ is calculated based on the dot product of E and E^T. After that, when a user-specified target is provided, the nearest neighboring words of the target will be returned as t-words, based on their similarity values in M. Notice that other semantics learning models can be used in a similar way [4]. Probabilistic topic models [2] can be used as well, by searching the corresponding topic for the given target and returning the topical words. Notice that this stage is highly similar to the (unsupervised) target extraction in sentiment analysis [9] and many existing models can be utilized at this stage. The key difference is that, the target name is specified in our setting, so we do not have to perform a full extraction of all possible targets covered by a given corpus, but only focus on the given target (name) by returning its nearest neighbors.

4 Stage Two: Disentangling

4.1 PU Learning using Word Vectors

This stage separates the given t-words into *t-aspect words* and *t-opinion words*. As discussed in Section 1, in order to provide a general approach without manual labeling for any possible domain, we formulate this problem as a binary classification task in a PU learning setting [8]. Clearly, in addition to aspect and opinion words, a domain vocabulary also contains other words like background words.

However, as indicated in [14, 18], those words do not have a seriously bad effect as they are unlikely to be semantically similar to a given target. Therefore, we assume/treat most of the non-opinion words in the t-words are/as aspect words. This assumption holds well as shown in studies [14, 18] and our experiments.

In regard to PU learning, it can be understood as a particular type of semi-supervised learning methods, which learns a binary classifier using only positive and unlabeled examples (with no negative examples). Here P represents a set of data examples with positive labels. In our task, the opinion words from an opinion lexicon will be the words in P, such as "good", "bad" and "angry". In terms of U, it denotes the set of data examples with unknown labels. In our case, other words that are not in the lexicon are in U., where U consists of both (true) opinion words and non-opinion words. With word vectors as features and a set of general opinion words as positive labels, we can build a PU classifier. In our work, we use logistic regression[2] as the PU classifier, which can generate the probabilistic score of a word being in the positive class (i.e., opinion word). In this way, words from U with high prediction scores can be detected as newly identified opinion words, and we can identify more words iteratively with new opinion words being found. However, a notable issue in this PU setting is that the errors from false positive (FP) examples (wrongly predicted opinion words) can be propagated, thus degenerating its performance. To address it, we exploit the idea of *lifelong machine learning* [3] and incorporate it into PU learning. The idea is to exploit the classification knowledge learned from past domains to increase the correctness and reliability of the newly detected opinion words.

4.2 Lifelong Machine Learning

Lifelong machine learning [3] or *lifelong learning*/LL, works by retaining the knowledge learned from the past tasks and uses it to help future learning, i.e., to help the current or incoming task. It mimics how we humans learn. With regard to sentiment analysis, we (human beings) can learn many opinion expressions in our lives across different domains/products, which enables us to better understand and identify opinion words in a new domain. More details can be found in [3]. Following the LL fashion, our system retains the newly learned opinion words every time it has finished processing one domain (one task), treating them as knowledge and accumulating them. The system accumulates such knowledge continuously from continuous domain/product learning. So in any time it has processed N domains and starts to process the $(N+1)$th domain, the accumulated knowledge will be used to help generate more reliable opinion words that are suitable for the $(N+1)$th domain. Based on this general idea, we develop a novel lifelong PU (LPU) learning algorithm.

4.3 Lifelong PU learning (LPU)

Our proposed LPU algorithm consists of four main steps, which are detailed as follows. The overall algorithm is given in Algorithm 1 (Alg. 1).

Algorithm 1 Lifelong PU (LPU) Learning

Input: Current domain corpus $D_{i=n+1}$, Past domain corpora $\boldsymbol{D}=\{D_1, ..D_j .., D_n\}$
Opinion words in lexicon W^P, Maximum learning iteration m
Number of learned words in one iteration l
Output: All newly-extracted opinion words W_i^+ in D_i

1: // Step 1. Knowledge Accumulation
2: **for** each domain corpus $D_j \in \boldsymbol{D}$ **do**
3: $W_j, V_j \leftarrow GetWordsAndEmbeddings(D_j)$
4: $W_j^P \leftarrow W_j \cap W^P$, $W_j^U \leftarrow W_j - W_j^P$
5: $c_j \leftarrow PUClassifier(V_j, W_j^P)$
6: $W_j^+ \leftarrow OpinionWordPrediction(c_j, W_j^U)$
7: $\boldsymbol{SKB} \leftarrow \boldsymbol{SKB} \cup W_j^+$ // sentiment knowledge base
8: **end for**
9: // Step 2. Current Domain Setup
10: $W_i, V_i \leftarrow GetWordsAndEmbeddings(D_i)$
11: $W_i^P \leftarrow W_i \cap W^P$, $W_i^U \leftarrow W_i - W_i^P$
12: Create a hash-table H to store top neighbors of all words
13: $W_i^{RN} \leftarrow GetReliableNeighbors(H, W_i^P)$
14: // Step 3. Knowledge Mining and Preparation
15: $W^{SK} \leftarrow FIM(\boldsymbol{SKB})$
16: $W_i^{SK} \leftarrow W_i \cap W^{SK}$ // sentiment-knowledge for domain i
17: $W_i^{RS} = \varnothing$ // reliable learned opinion words
18: $W_i^{PP} = \varnothing$ // current positive prediction (opinion words)
19: $W_i^{NS} = W_i^{RN} \cap W_i^{SK}$ // newly learned sentiment
20: // Step 4. Restricted PU Iterations
21: $t = 0$
22: **while** $t < m$ or W_i^{NS} is not empty **do**
23: $W_i^{RS} \leftarrow W_i^{RS} \cup W_i^{NS}$ // updating reliable sentiment
24: $c_i \leftarrow PUClassifier(V_i, W_i^{RS} \cup W_i^P)$
25: $W_i^{NEW1} \leftarrow MineReliableOpinion(W_i^{SK}, W_i^{PP}, H, l)$
26: $W_i^{NEW2} \leftarrow ReliableOpinion(W_i^{PP}, W_i^{PP}, H, l)$
27: $W_i^{NS} \leftarrow W_i^{NEW1} \cup W_i^{NEW2}$
28: $W_i^{PP} \leftarrow OpinionWordPrediction(c_i, V_i)$
29: $t = t + 1$
30: **end while**
31: $W_i^+ \leftarrow OpinionWordPrediction(c_i, W_i^U)$

Algorithm 2 *MineReliableOpinion*(A, B, H, l)

1: $S = \varnothing$ // counts positive neighbors for every word in A
2: **for** each a word $w \in A$ **do**
3: $S \leftarrow countPositiveNeighbors(B, H(w))$
4: **end for**
5: return $sortAndReturnTopLWords(A, S, l)$

Step 1: Knowledge Accumulation (lines 1-8) This step follows the traditional classification process but with knowledge retention for building a knowledge base from past domains. Specifically, for each domain (task j), we first obtain its vocabulary W_j and semantic representation of words V_j (line 3). With a general opinion lexicon, we then have the lexicon-based opinion words W_j^P, i.e., positive examples, and unlabeled examples W_j^U (line 4). A PU classifier is trained (line 5) and used to predict the probabilistic class scores of words in

[2] Other similar models can be also used in the same way.

W_j^U and to find new opinion words W_j^+ (line 6). After that, we retain W_j^+ as knowledge for constructing a knowledge base SKB (line 7).

Step 2: Current Domain Setup (lines 9-13) This step is for the current domain processing setup. The vocabulary words W_i and their semantic representation V_i, lexicon-based opinion words (positive examples) W_i^p and unlabeled words W_i^U of the current domain are first created (lines 10-11). Then we build a hash table H to store the nearest neighbors[3] for all words, which can be easily constructed from the similarity matrix M (see Section 3). With the table H established (line 12, and it is a one-time effort), the similarity query becomes a lookup operation. This H not only helps in the current step 2, but also plays a crucial role in the following step 4, as we will see shortly. Based on H, we can find the nearest neighbors for the lexicon-based opinion words and we call them reliable neighbors (line 13). This is an initial constraint, which is also intuitive, as those unlabeled/candidate words which are highly similar to the opinion words known from a lexicon should be more reliable (as opinion words).

Step 3: Knowledge Mining and Preparation (lines 14-19). This step is for mining knowledge and making preparation for later use. With the knowledge accumulated from many past domains and stored in SKB, we can extract the reliable knowledge W^{SK} (line 15). Here we adopt frequent itemset mining (FIM) [1]. The rationale behind it is that - the candidate words frequently predicted as opinion words in many different domains are more trustworthy and confident to be the real opinion words. The intersection of the reliable neighbors W_i^{RN} and reliable knowledge W_i^{SK} initializes W_i^{NS}, the newly learned sentiment (line 19). Lines 16-18 define other variables that are used in step 4, where W_i^{SK} denotes the sentiment knowledge for current domain i, W_i^{RS} the reliable learned sentiment (opinion words) during the PU learning iteration, and W_i^{PP} the newly-predicted opinion words in an ongoing iteration.

Step 4: Restricted PU Iterations (lines 21-31). This step performs iterative PU learning with imposed constraints. Unlike self-bootstrapping methods, here the expansion of the newly-predicted opinion words as positive examples in LPU is controlled more strictly. Only the reliable ones could be further used. The initialized new opinion words W_i^{NS} have already been restricted (see step 2) and used as initial reliable sentiment W_i^{RS}. During the iterative learning process, it keeps being updated (line 23) by adding only reliable opinion words (line 27). We develop two ways for expanding new reliable opinion words. One way is to learn from the reliable knowledge (line 25) and another way is to learn from its self-predicted results (line 26). Both ways are restricted by the defined reliability score shown in Algorithm 2 (Alg. 2). This score is calculated based on the number of identified positive neighbors of a candidate word, which is also used for ranking. In Alg. 2, A denotes the candidate word set and B denotes positive examples. The identified positive neighbors are from the intersection of positive examples (provided by B) and the neighbors of a candidate word (provide by $H(w)$). In each iteration, only the top l ranked words will be trusted and added as new positive examples. When the maximum iteration is met or there are no

[3] Simply using top 10 neighbors works consistently well for different domains.

more new opinion words that the system can learn, the iterative learning process stops and all newly-detected opinion words are returned (line 31).

5 Experiments

5.1 Candidate Methods for Comparison

Adjective Extraction (ADJ): This baseline regards all adjective words as opinion words and others as aspect words. This is a simple but widely used solution. We used POS tagging to extract adjectives. No classifier is trained here.
Part-Of-Speech (POS): The POS features have been reported effective for aspect and opinion extractions. This is also a representative syntax-based approach used in many related works [14, 18]. Here every word is represented by the POS features of its context, i.e., w_i will be represented as $[POS_{i-1}, POS_i, POS_{i+1}]$. This is used as the word representation for building a classifier.
Latent Semantic Indexing (LSI): LSI is a standard matrix factorization technique to construct latent semantic vectors. Its factorized word-feature correlation matrix can be used as word vector representation [15] to build a classifier.
Latent Dirichlet Allocation (LDA): LDA [2] is a classic topic model which discovers hidden topics from documents and groups words into topics. Similar to LSI, the term-topic matrix is used as the word vector representation [10].
Non-Lifelong Learning (NLL): This method is based on our introduced solution but without lifelong learning. It uses the word vectors learned by neural word embeddings to build a classifier.
Lifelong PU (LPU): This is our proposed lifelong PU learning algorithm introduced in Alg. 1.
Lifelong PU minor (LPU-): This is an LPU variant that does not make self-prediction explorations and relies more on the past mined knowledge. In other words, it considers the first type of reliable sentiment only (lines 25 in Alg. 1). This can be viewed as a conservative version of LPU.

5.2 Experimental Setup

Data. We use a large corpus of Amazon reviews from 20 different domains provided by [11] and the full list is shown in Table 1. For training all PU classifiers, a general opinion lexicon [6] is used so the words appear in it are automatically labeled as P. For testing/evaluation, we manually label the aspect and opinion words. Specifically, three domains from different products are selected, namely, *cellphone*, *beauty* and *office*. For each domain, three different targets are specified for evaluation. More details are given in Table 1.
Parameters and Settings. For every candidate method except ADJ, their word vectors/features are learned and used for classification. Specifically, for LSI and LDA, we obtained the term-feature matrix and term-topic matrix. For NLL, LPU- and LPU, we used the skip-gram model [13]. The vector dimension is set to 200 as default and we maintain the same size for LDA and LSI. Logistic regression is used as the classifier for all methods. For LPU, we treat other 19 domains

Dataset	#Reviews	#Words	Words in Lexicon	Target for Evaluation
CellPhone	194,439	28,942	2,764	display, volumes, weight
Beauty	198,502	29,695	2,778	cleansers, fragrance, groomers
Office	53, 258	20,858	2,332	papers, clips, chairs
Full domains	apps for android, amazon instance video, automotive, baby, grocery, health, kindle, tools/home improvement, home and kitchen			

Table 1: Detailed information about the domains for evaluation and the full list.

besides the current domain as the past domains to mine knowledge. Notice that for a current domain, only its domain data and the automatically accumulated knowledge will be used, and no other extra domain data will be available, which follows the lifelong learning experimental setting from existing works [18, 3]. We empirically set the minimum support to 5 for frequent opinion word mining. We set the maximum iterations m to 10 and the number of words l to learn in each iteration to 50. They work consistently well on different domains/datasets.

5.3 Quantitative Evaluation

Accuracy is used as the metric for evaluation because our task is formulated as a binary classification problem. However, since it is hard to know the exact number of all related words (t-words) to a given target, we use the accuracy@n (acc@n) as our evaluation measure, where n is set to 50, 100, and 150. Specifically, given a target, we first collect its top t-words, and manually label them as opinion or aspect words.

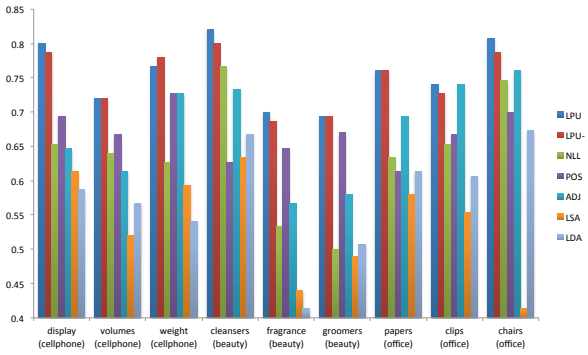


Fig. 2: Acc@150 for all models and targets

We then apply every trained candidate model to classify those top n words to calculate its corresponding acc@n. The results are reported in Figure 2, 3 and 4. Based on them, we have the following observations:

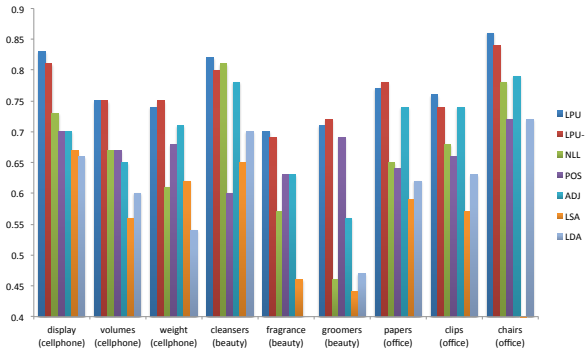


Fig. 3: Acc@100 for all models and targets

(1) LPU and LPU- outperform other baselines markedly. LPU improves the best baseline

results by 8.29%, 7.11% and 4.00% in acc@150, acc@100, and acc@50. Likewise, LPU- improves the best baseline results by 7.55%, 6.44% and 5.77% in acc@150, acc@100, and acc@50. They demonstrate the effectiveness of lifelong learning.

(2) LPU achieves better performance than LPU- in acc@150, acc@100 but is inferior to LPU- in acc@50. This indicates that LPU is more accurate by considering a big n (more t-words), but LPU- could be more suitable if we only focus on the top-ranked words.

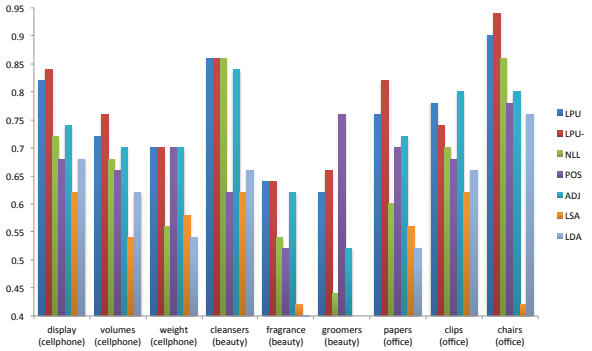


Fig. 4: Acc@50 for all models and targets

(3) Among other baselines, we observe that NLL and POS perform the best. While POS explicitly reflects the contextual syntax, it is worth noting that the word embeddings used in NLL is also implicitly learned from word-context matrix [7], which implies the syntactic information is very useful in this task.

Target: Volumes (Domain: CellPhone)		
Model	**Aspect**	**Opinion**
LPU	volumes, bass, *undistorted*, volume, Gaga, pitches, sound, *harshness*, cymbals, treble, eq, conf,	muddiness, shrill, trebles, distortion, hissy, loud, sibilance, thumping, thump, highs, soundstage, midrange,
LPU-	bass, volume, Gaga, pitches, sound, cymbals, treble, conf, Mids, reproduces, bitrate, Highs,	*volumes*, muddiness, undistorted, shrill, trebles, distortion, hissy, loud, sibilance, harshness, thumping, eq,
NLL	volumes, *muddiness*, bass, *undistorted*, trebles, volume, Gaga, pitches, sound, *harshness*, cymbals, treble,	shrill, distortion, hissy, loud, sibilance, midrange, *equalization*, tinny, distorted, piercing, muddy, louder,
POS	volumes, *muddiness*, trebles, Gaga, pitches, *sibilance*, cymbals, Mids, LiveAudio, highs, *reproducing*, Highs,	*bass*, undistorted, shrill, distortion, hissy, *volume*, loud, *sound*, harshness, *treble*, thumping, *eq*,
ADJ	volumes, *muddiness*, bass, trebles, volume, Gaga, pitches, sound, *sibilance*, *harshness*, cymbals, *thumping*,	undistorted, shrill, distortion, hissy, loud, *treble*, loudest, tinny, distorted, *Treble*, resonant, muddy,

Table 2: Incorrect sentiment words are italicized and marked in red.

5.4 Qualitative Evaluation

This subsection shows some example results in Table 2. Since LSI and LDA have much poorer performances than others, we do not include their results here. The represented words are the top predicted aspect words and top predicted opinion words from the t-words of a given target. Incorrect opinion words are italicized and marked in red. As we can see, LPU and LPU- better distinguish aspect and opinion words. For example, the opinion word "muddiness" is extracted by them but not by other models. POS identifies many wrong opinion words like "sound"

Topic: Skin (Domain: Beauty)	
Aspect	**Newly-Identified Opinion**
face, skin, use, acne (-b), using, just (-b), wash, feel (-b), make, day, product (-b), lotion	dry (-c,d), rid (-c,d), oily (-c,d), drying (-a,c,d), mild (-c,d), notice (-c,d), huge (-d), new (-c,d), ok (-c,d), younger (-c)
Topic: Headset (Domain: CellPhone)	
Aspect	**Newly-Identified Opinion**
headset, sound, quality (-b), bluetooth, adapter, hear (-b), ear, volume, headsets, way (-b)	really (-c,d), long (-a), low (-d), away (-c,d), high (-d), short, quite (-c,d), ok, idea (-c,d), close (-c,d)

Table 3: "-" symbol indicates the models behind it do not identify the word.

and "volume". Although ADJ is good at extracting adjective opinion words, it misses other opinion words like "muddiness", "sibilance" and "thumping". NLL also misses many opinion words like ADJ.

5.5 Further Analysis

In order to further evaluate the generality of our proposed approach, we applied it to another popular target extraction technique - topic modeling. Specifically, we run LDA [2] for topic generation and then use our algorithm to separate aspect words and opinion words. It also produces reasonably good results as shown in Table 3. Notations in "-(a, b, c, d)" will be explained below.

We also investigated the effect of alleviating FP error propagation in LPU. We denote three types of iterative-learning models as (a), (b), and (c), and they learned with 10 iterations by considering their newly-identified opinion words as positive examples: (a) LPU, using Alg. 1; (b) A PU model selecting its predicted positive examples ($prob > 0.5$) in the current iteration as P for the next iteration; (c) A PU model that always combines the newly-predicted positive examples with the initial lexicon-based positive examples, without using the constraints in LPU. We also denote NLL as the model (d), which does not learn iteratively.

We now take a further look at Table 3. The "-" symbol indicates that the models following it do not identify the word. Note that here the opinion words from lexicon P are excluded so we can see how those models perform in classifying the unlabeled words U. We observe: 1). Model (c) misses many interesting opinion words like model (d), which indicates that its positive examples remain very similar during all iterations, i.e., it does not learn many new positive examples; 2). Model (b) mis-classifies many aspect words as opinion words as its FP errors propagate iteratively, i.e., the model is confused by the newly-added false positive examples; 3). Model (a), which is LPU, works robustly well.

6 Conclusion

This paper discussed the problem of disentangling t-opinion words and t-aspect words from the grouped t-words for fine-grained target-based sentiment analysis (FTSA). We formulated this problem in a PU learning setting and incorporated the lifelong learning idea to overcome the drawback of error propagation in PU

learning. To achieve this, a novel lifelong PU learning (LPU) model was proposed. Our experimental results using real-world data demonstrated its effectiveness.

Acknowledgments

This work was partially supported by a grant from the National Science Foundation (NSF IIS 1838770) and a research gift from Northrop Grumman Mission Systems.

References

1. Agrawal, R., Srikant, R., et al.: Fast algorithms for mining association rules. In: VLDB. vol. 1215, pp. 487–499 (1994)
2. Blei, D.M., Ng, A.Y., Jordan, M.I.: Latent dirichlet allocation. Journal of machine Learning research **3**(Jan), 993–1022 (2003)
3. Chen, Z., Liu, B.: Lifelong machine learning. Synthesis Lectures on Artificial Intelligence and Machine Learning **10**(3), 1–145 (2016)
4. Deerwester, S., Dumais, S.T., Furnas, G.W., Landauer, T.K., Harshman, R.: Indexing by latent semantic analysis. Journal of the American society for information science **41**(6), 391 (1990)
5. Harris, Z.S.: Distributional structure. Word **10**(2-3), 146–162 (1954)
6. Hu, M., Liu, B.: Mining and summarizing customer reviews. In: KDD. pp. 168–177. ACM (2004)
7. Levy, O., Goldberg, Y., Dagan, I.: Improving distributional similarity with lessons learned from word embeddings. TACL **3**, 211–225 (2015)
8. Li, X.L., Liu, B.: Learning from positive and unlabeled examples with different data distributions. In: ECML. pp. 218–229. Springer (2005)
9. Liu, B.: Sentiment analysis and opinion mining. Synthesis lectures on human language technologies **5**(1), 1–167 (2012)
10. Maas, A.L., Daly, R.E., Pham, P.T., Huang, D., Ng, A.Y., Potts, C.: Learning word vectors for sentiment analysis. In: ACL. pp. 142–150 (2011)
11. McAuley, J., Pandey, R., Leskovec, J.: Inferring networks of substitutable and complementary products. In: KDD. pp. 785–794. ACM (2015)
12. Mignone, P., Pio, G.: Positive unlabeled link prediction via transfer learning for gene network reconstruction. In: ISMIS. pp. 13–23. Springer (2018)
13. Mikolov, T., Dean, J.: Distributed representations of words and phrases and their compositionality. NIPS (2013)
14. Mukherjee, A., Liu, B.: Aspect extraction through semi-supervised modeling. In: ACL. pp. 339–348. ACM (2012)
15. Pu, X., Jin, R., Wu, G., Han, D., Xue, G.R.: Topic modeling in semantic space with keywords. In: CIKM. pp. 1141–1150. ACM (2015)
16. Tversky, A.: Features of similarity. Psychological review **84**(4), 327 (1977)
17. Vo, D.T., Zhang, Y.: Target-dependent twitter sentiment classification with rich automatic features. In: IJCAI. pp. 1347–1353 (2015)
18. Wang, S., Chen, Z., Liu, B.: Mining aspect-specific opinion using a holistic lifelong topic model. In: WWW. pp. 167–176. WWW (2016)
19. Wang, S., Lv, G., Mazumder, S., Fei, G., Liu, B.: Lifelong learning memory networks for aspect sentiment classification. In: Big Data 2018. pp. 861–870 (2018)
20. Yang, Z., Yang, D., Dyer, C., He, X., Smola, A., Hovy, E.: Hierarchical attention networks for document classification. In: NAACL. pp. 1480–1489 (2016)