Lifelong-RL: Lifelong Relaxation Labeling for Separating Entities and Aspects in Opinion Targets

Lei Shu[1], **Bing Liu**[1], **Hu Xu**[1], **Annice Kim**[2]
[1]Department of Computer Science, University of Illinois at Chicago, USA
[2]Center for Health Policy Science and Tobacco Research, RTI International, USA
[1]{lshu3, liub, hxu48}@uic.edu, [2]akim@rti.org

Abstract

It is well-known that opinions have targets. Extracting such targets is an important problem of opinion mining because without knowing the target of an opinion, the opinion is of limited use. So far many algorithms have been proposed to extract opinion targets. However, an opinion target can be an entity or an aspect (part or attribute) of an entity. An opinion about an entity is an opinion about the entity as a whole, while an opinion about an aspect is just an opinion about that specific attribute or aspect of an entity. Thus, opinion targets should be separated into entities and aspects before use because they represent very different things about opinions. This paper proposes a novel algorithm, called *Lifelong-RL*, to solve the problem based on *lifelong machine learning* and *relaxation labeling*. Extensive experiments show that the proposed algorithm Lifelong-RL outperforms baseline methods markedly.

1 Introduction

A core problem of opinion mining or sentiment analysis is to identify each opinion/sentiment target and to classify the opinion/sentiment polarity on the target (Liu, 2012). For example, in a review sentence for a car, one wrote "Although the engine is slightly weak, this car is great." The person is positive (opinion polarity) about the car (opinion target) as a whole, but slightly negative (opinion polarity) about the car's engine (opinion target).

Past research has proposed many techniques to extract *opinion targets* (we will just call them *targets* hereafter for simplicity) and also to classify sentiment polarities on the targets. However, a target can be an entity or an aspect (part or attribute) of an entity. "Engine" in the above sentence is just one aspect of the car, while "this car" refers to the whole car. Note that in (Liu, 2012), an entity is called a *general aspect*. For effective opinion mining, we need to classify whether a target is an entity or an aspect because they refer to very different things. One can be positive about the whole entity (car) but negative about some aspects of it (e.g., engine) and vice versa. This paper aims to perform the target classification task, which, to our knowledge, has not been attempted before. Although in supervised extraction one can annotate entities and aspects with separate labels in the training data to build a model to extract them separately, in this paper our goal is to help unsupervised target extraction methods to classify targets. Unsupervised target extraction methods are often preferred because they save the time-consuming data labeling or annotation step for each domain.

Problem Statement: Given a set of opinion targets $T = \{t_1, \ldots, t_n\}$ extracted from an opinion corpus d, we want to classify each target $t_i \in T$ into one of the three classes, *entity*, *aspect*, or *NIL*, which are called class labels. NIL means that the target is neither an entity nor an aspect and is used because target extraction algorithms can make mistakes.

This paper does not propose a new target extraction algorithm. We use an existing unsupervised method, called *Double Propagation* (DP) (Qiu et al., 2011), for extraction. We only focus on target classification after the targets have been extracted. Note that an entity here can be a named entity, a prod-

uct category, or an abstract product (e.g., "this machine" and "this product"). An named entity can be the name of a brand, a model, or a manufacturer. An aspect is a part or attribute of an entity, e.g., "battery" and "price" of the entity "camera".

Since our entities not just include the traditional named entities (e.g., "Microsoft" and "Google") but also other expressions that refer to such entities, traditional named entity recognition algorithms are not sufficient. Pronouns such as "it," "they," etc., are not considered in this paper as co-reference resolution is out of the scope of this work.

We solve this problem in an unsupervised manner so that there is no need for labor-intensive manual labeling of the training data. One key observation of the problem is that although entities and aspects are different, they are closely related because aspects are parts or attributes of entities and they often have syntactic relationships in a sentence, e.g., "This phone's screen is super." Thus it is natural to solve the problem using a relational learning method. We employ the graph labeling algorithm, *Relaxation Labeling* (RL) (Hummel and Zucker, 1983), which performs unsupervised belief propagation on a graph. In our case, each target extracted from the given corpus d forms a graph node and each relation identified in d between two targets forms an edge. With some initial probability assignments, RL can assign each target node the most probable class label. Although some other graph labeling methods can be applied as well, the key issue here is that just using a propagation method in isolation is far from sufficient due to lack of information from the given corpus, which we detail in Section 5. We then employ *Lifelong Machine Learning* (LML) (Thrun, 1998; Chen and Liu, 2014b) to make a major improvement.

LML works as follows: The learner has performed a number learning tasks in the past and has retained the knowledge gained so far. In the new/current task, it makes use of the past knowledge to help current learning and problem solving. Since RL is unsupervised, we can assume that the system has performed the same task on reviews of a large number of products/domains (or corpora). It has also saved all the graphs and classification results from those past domains in a *Knowledge Base* (KB). It then exploits this past knowledge to help classification in the current task/domain. We call this combined approach of relaxation labeling and LML *Lifelong-RL*. The approach is effective because there is a significant amount of sharing of targets and target relations across domains.

LML is different from the classic learning paradigm (supervised or unsupervised) because classic learning has no memory. It basically runs a learning algorithm on a given data in isolation without considering any past learned knowledge (Silver et al., 2013). LML aims to mimic human learning, which always retains the learned knowledge from the past and uses it to help future learning.

Our experimental results show that the proposed Lifelong-RL system is highly promising. The paradigm of LML helps improve the classification results greatly.

2 Related Work

Although many target extraction methods exist (Hu and Liu, 2004; Zhuang et al., 2006; Ku et al., 2006; Wang and Wang, 2008; Wu et al., 2009; Lin and He, 2009; Zhang et al., 2010; Mei et al., 2007; Li et al., 2010; Brody and Elhadad, 2010; Wang et al., 2010; Mukherjee and Liu, 2012; Fang and Huang, 2012; Zhou et al., 2013; Liu et al., 2013; Poria et al., 2014), we are not aware of any attempt to solve the proposed problem. As mentioned in the introduction, although in supervised target extraction, one can annotate entities and aspects with different labels, supervised methods need manually labeled training data, which is time-consuming and labor-intensive to produce (Jakob and Gurevych, 2010; Choi and Cardie, 2010; Mitchell et al., 2013). Note that relaxation labeling was used for sentiment classification in (Popescu and Etzioni, 2007), but not for target classification. More details of opinion mining can be found in (Liu, 2012; Pang and Lee, 2008).

Our work is related to transfer learning (Pan and Yang, 2010), which uses the source domain labeled data to help target domain learning, which has little or no labeled data. Our work is not just using a source domain to help a target domain. It is a continuous and cumulative learning process. Each new task can make use of the knowledge learned from all past tasks. Knowledge learned from the new task can also help improve learning of any past task. Transfer learning is not continuous, does not

accumulate knowledge over time and cannot improve learning in the source domain. Our work is also related to multi-task learning (Caruana, 1997), which jointly optimizes a set of related learning tasks. Clearly, multi-task learning is different as we learn and save information which is more realistic when a large number of tasks are involved.

Our work is most related to Lifelong Machine Learning (LML). Traditional LML focuses on supervised learning (Thrun, 1998; Ruvolo and Eaton, 2013; Chen et al., 2015). Recent work used LML in topic modeling (Chen and Liu, 2014a), which is unsupervised. Basically, they used topics generated from past domains to help current domain model inference. However, they are just for aspect extraction. So is the method in (Liu et al., 2016). They do not solve our problem. Their LML methods are also different from ours as we use a graph and results obtained in the past domains to augment the current task/domain graph to solve the problem.

3 Lifelong-RL: The General Framework

In this section, we present the proposed general framework of lifelong relaxation labeling (Lifelong-RL). We first give an overview of the relaxation labeling algorithm, which forms the base. We then incorporate it with the LML capability. The next two sections detail how this general framework is applied to our proposed task of separating entities and aspects in opinion targets.

3.1 Relaxation Labeling

Relaxation Labeling (RL) is an unsupervised graph-based label propagation algorithm that works iteratively. The graph consists of nodes and edges. Each edge represents a binary relationship between two nodes. Each node t_i in the graph is associated with a multinomial distribution $P(L(t_i))$ ($L(t_i)$ being the label of t_i) on a label set Y. Each edge is associated with two conditional probability distributions $P(L(t_i)|L(t_j))$ and $P(L(t_j)|L(t_i))$, where $P(L(t_i)|L(t_j))$ represents how the label $L(t_j)$ influences the label $L(t_i)$ and vice versa. The neighbors $Ne(t_i)$ of a node t_i are associated with a weight distribution $w(t_j|t_i)$ with $\sum_{t_j \in Ne(t_i)} w(t_j|t_i) = 1$.

Given the initial values of these quantities as inputs, RL iteratively updates the label distribution of each node until convergence. Initially, we have $P^0(L(t_i))$. Let $\Delta P^{r+1}(L(t_i))$ be the change of $P(L(t_i))$ at iteration $r+1$. Given $P^r(L(t_i))$ at iteration r, $\Delta P^{r+1}(L(t_i))$ is computed by:

$$\Delta P^{r+1}(L(t_i)) = \sum_{t_j \in Ne(t_i)} (w(t_j|t_i) \cdot \sum_{y \in Y} (P(L(t_i)|L(t_j) = y) P^r(L(t_j) = y))) \quad (1)$$

Then, the updated label distribution for iteration $r+1$, $P^{r+1}(L(t_i))$, is computed as follows:

$$P^{r+1}(L(t_i)) = \frac{P^r(L(t_i))(1+\Delta P^{r+1}(L(t_i)))}{\sum_{y \in Y} P^r(L(t_i)=y)(1+\Delta P^{r+1}(L(t_i)=y))} \quad (2)$$

Once RL ends, the final label of node t_i is its highest probable label: $L(t_i) = \underset{y \in Y}{\operatorname{argmax}}(P(L(t_i) = y))$.

Note that $P(L(t_i)|L(t_j))$ and $w(t_j|t_i)$ are not updated in each RL iteration but only $P(L(t_i))$ is. $P(L(t_i)|L(t_j))$, $w(t_j|t_i)$ and $P^0(L(t_i))$ are provided by the user or computed based on the application context. RL uses these values as input and iteratively updates $P(L(t_i))$ based on Equations (1) and (2) until convergence. Next we discuss how to incorporate LML in RL.

3.2 Lifelong Relaxation Labeling

For LML, it is assumed that at any time step, the system has worked on u past domain corpora $D = \{d_1, \ldots, d_u\}$. For each past domain corpus $d \in D$, the same Lifelong-RL algorithm was applied and its results were saved in the Knowledge Base (KB). Then the algorithm can borrow some useful prior/past knowledge in the KB to help RL in the new/current domain d_{u+1}. Once the results of the current domain are produced, they are also added to the KB for future use.

We now detail the specific types of information or knowledge that can be obtained from the past domains to help RL in the future, which should thus be stored in the KB.

1. *Prior edges*: In many applications, the graph is not given. Instead, it has to be constructed based on the data from the new task/domain data d_{u+1}. However, due to the limited data in d_{u+1}, some edges between nodes that should be present are not extracted from the data. But such edges between the nodes may exist in

some past domains. Then, those edges and their associated probabilities can be borrowed.

2. *Prior labels*: Some nodes in the current new domain may also exist in some past domains. Their labels in the past domains are very likely to be the same as those in the current domain. Then, those prior labels can give us a better idea about the initial label probability distributions of the nodes in the current domain d_{u+1}.

To leverage those edges and labels from the past domains, the system needs to ensure that they are likely to be correct and applicable to the current task domain. This is a challenge problem. In the next two sections, we detail how to ensure these to a large extent in our application context along with how to compute those initial probabilities.

4 Initialization of Relaxation Labeling

We now discuss how the proposed Lifelong-RL general framework is applied to solve our problem. In our case, each node in the graph is an extracted target $t_i \in T$, and each edge represents a binary relationship between two targets. T is the given set of all opinion targets extracted by an extraction algorithm from a review dataset/corpus d. The label set for each target is $Y = \{\text{entity}, \text{aspect}, \text{NIL}\}$. In this section, we describe how to use text clues in the corpus d to compute $P(L(t_i)|L(t_j))$, $w(t_j|t_i)$ and $P^0(L(t_i))$. In the next section, we present how these quantities are improved using prior knowledge from the past domains in the LML fashion.

4.1 Text Clues for Initialization

We use two kinds of text clues, called *type modifiers* $M(t)$ and *relation modifiers* M_R to compute the initial label distribution $P(L(t_i))$ and conditional label distribution $P(L(t_i)|L(t_j))$ respectively.

Type Modifier: This has two kinds $M_T = \{m_E, m_A\}$, where m_E and m_A represent entity modifier and aspect modifier respectively. For example, the word "this" as in "this camera is great" indicates that "camera" is probably an entity. Thus, "this" is a type modifier indicating $M(camera) = m_E$. "These" is also a type modifier. Aspect modifier is implicitly assumed when the number of appearances of entity modifiers is less than or equal to a threshold (see Section 4.2).

Relation Modifier: Given two targets, t_i and t_j, we use $M_{t_j}(t_i)$ to denote the relation modifier that the label of target t_i is influenced by the label of target t_j. Relation modifiers are further divided into 3 kinds: $M_R = \{m_c, m_{A|E}, m_{E|A}\}$.

Conjunction modifier m_c: Conjoined items are usually of the same type. For example, in "price and service", "and service" indicates a conjunction modifier for "price" and vice versa.

Entity-aspect modifier $m_{A|E}$: A possessive expression indicates an entity and an aspect relation. For example, in "the camera's battery", "camera" indicates an *entity-aspect modifier* for "battery".

Aspect-entity modifier $m_{E|A}$: Same as above except that "battery" indicates an *aspect-entity modifier* for "camera".

Modifier Extraction: These modifiers are identified from the corpus d using three syntactic rules. "This" and "these" are used to extract type modifier $M(t) = m_E$. $C_{m_E}(t)$ is the occurrence count of that modifier on target t, which is used in determining the initial label distribution in Section 4.2.

Relation modifiers are identified by dependency relations $conj(t_i, t_j)$ and $poss(t_i, t_j)$ using the Stanford Parser (Klein and Manning, 2003). Each occurrence of a relation rule contributes one count of $M_{t_j}(t_i)$ for t_i and one count of $M_{t_i}(t_j)$ for t_j. We use $C_{m_c,t_j}(t_i)$, $C_{m_{A|E},t_j}(t_i)$ and $C_{m_{E|A},t_j}(t_i)$ to denote the count of t_j modifying t_i with conjunction, entity-aspect and aspect-entity modifiers respectively. For example, "price and service" will contribute one count to $C_{m_c,\text{price}}(\text{service})$ and one count to $C_{m_c,\text{service}}(\text{price})$. Similarly, "camera's battery" will contribute one count to $C_{m_{A|E},\text{camera}}(\text{battery})$ and one count to $C_{m_{E|A},\text{battery}}(\text{camera})$.

4.2 Computing Initial Probabilities

The initial label probability distribution of target t is computed based on $C_{m_E}(t)$, i.e.,

$$P^0(L(t)) = \begin{cases} P_{m_E}(L(t)) & \text{if} \quad C_{m_E}(t) > \alpha \\ P_{m_A}(L(t)) & \text{if} \quad C_{m_E}(t) \leq \alpha \end{cases} \tag{3}$$

Here, we have two pre-defined distributions: P_{m_E} and P_{m_A}, which have a higher probability on entity and aspect respectively. The parameter α is a threshold indicating that if the entity modifier rarely occurs, the target is more likely to be an aspect. These

values are set empirically (see Section 6).

Let term $q(M_{t_j}(t_i) = m)$ be the normalized weight on the count for each kind of relation modifier $m \in M_R$:

$$q(M_{t_j}(t_i) = m) = \frac{C_{m,t_j}(t_i)}{C_{t_j}(t_i)} \tag{4}$$

where $C_{t_j}(t_i) = \sum_{m \in M_R} C_{m,t_j}(t_i)$.

The conditional label distribution $P(L(t_i)|L(t_j))$ of t_i given the label of t_j is the weighted sum over the three kinds of relation modifiers:

$$\begin{aligned} &P(L(t_i)|L(t_j)) = \\ &q(M_{t_j}(t_i) = m_c) \cdot P_{m_c}(L(t_i)|L(t_j)) \\ &+q(M_{t_j}(t_i) = m_{A|E}) \cdot P_{m_{A|E}}(L(t_i)|L(t_j)) \\ &+q(M_{t_j}(t_i) = m_{E|A}) \cdot P_{m_{E|A}}(L(t_i)|L(t_j)) \end{aligned} \tag{5}$$

where P_{m_c}, $P_{m_{A|E}}$, and $P_{m_{E|A}}$ are pre-defined conditional distributions. They are filled with values to model the label influence from neighbors and can be found in Section 6.

Finally, target t_i's neighbor weight for target t_j, i.e., $w(t_j|t_i)$, is the ratio of the count of relation modifiers $C_{t_j}(t_i)$ over the total of all t_i's neighbors:

$$w(t_j|t_i) = \frac{C_{t_j}(t_i)}{\sum_{t_{j'} \in Ne(t_i)} C_{t_{j'}}(t_i)} \tag{6}$$

If $C_{t_j}(t_i) = 0$, t_i and t_j has no edge between them.

5 Using Past Knowledge in Lifelong-RL

Due to the fact that the review corpus d_{u+1} in the current task domain may not be very large and that we use high quality syntactic rules to extract relations to build the graph to ensure precision, the number of relations extracted can be small and insufficient to produce a graph that is information rich with accurate initial probabilities. We thus apply LML to help using knowledge learned in the past. The proposed LML process in Lifelong-RL for our task is shown in Figure 1.

Our prior knowledge includes type modifiers, relation modifiers and labels of targets obtained from past domains in D. Each record in the KB is stored as a 9-tuple: $(d, t_i, t_j, M^d(t_i), M^d(t_j), C^d_{m,t_j}(t_i), C^d_{m,t_i}(t_j), L^d(t_i), L^d(t_j))$ where $d \in D$ is a past domain; t_i and t_j are two targets; $M^d(t_i)$, $M^d(t_j)$ are their type

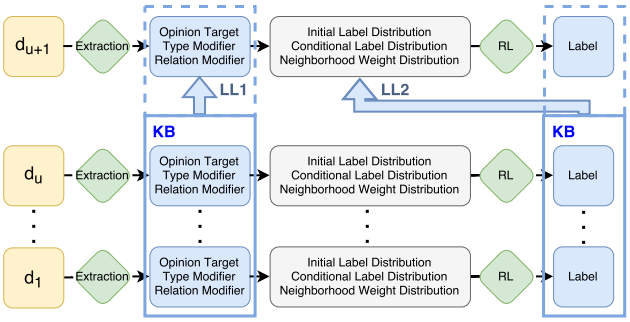

Figure 1: The proposed LML process.

modifiers, $C^d_{m,t_j}(t_i)$ and $C^d_{m,t_i}(t_j)$ are counts for relation modifiers; $L^d(t_i)$ and $L^d(t_j)$ are labels decided by RL. For example, the sentence "This camera's battery is good" forms: $(d, \text{camera}, \text{battery}, m_E, m_A, C_{m_{E|A},battery}(camera) = 1, C_{m_{A|E},camera}(battery) = 1, \text{entity}, \text{aspect})$. It means that in the past domain d, "camera" and "battery" are extracted targets. Since "camera" is followed by "this", its type modifier is m_E. Since "battery" is not identified by an entity modifier, it is m_A. The pattern "camera's battery" contributes one count for both relation modifiers $C_{m_{E|A},battery}(camera)$ and $C_{m_{A|E},camera}(battery)$. RL has labeled "camera" as entity and "battery" as aspect in d.

The next two subsections present how to use the knowledge in the KB to improve the initial assignments for the label distributions, conditional label distributions and neighborhood weight distributions in order to achieve better final labeling/classification results for the current/new domain d_{u+1}.

5.1 Exploiting Relation Modifiers in the KB

If two targets in the current domain corpus have no edge, we can check whether relation modifiers of the same two targets exist in some past domains. If so, we may be able to borrow them. But to ensure suitability, two consistency checks are performed.

Label Consistency Check: Since RL makes mistakes, we need to ensure that relation modifiers in a record in the KB are consistent with target labels in that past domain. For example, "camera's battery" is confirmed by "camera" being labeled as entity and "battery" being labeled as aspect in a past domain $d \in D$. Without this consistency, the record may not be reliable and should be discarded from the KB.

We define an indicator variable $\mathbb{I}^d_{m,t_j}(t_i)$ to en-

sure that the record r's relation modifier is *consistent* with the labels of its two targets:

$$\mathbb{I}^d_{m_{A|E},t_j}(t_i) = \begin{cases} 1 & \text{if } C^d_{m_{A|E},t_j}(t_i) > 0 \\ & \text{and } L^d(t_i) = \text{aspect} \\ & \text{and } L^d(t_j) = \text{entity} \\ 0 & \text{otherwise} \end{cases} \quad (7)$$

For example, if "camera" is labeled as entity and "battery" is labeled as aspect in the past domain d, we have $\mathbb{I}^d_{m_{A|E},\text{camera}}(\text{battery}) = 1$ and $\mathbb{I}^d_{m_{E|A},\text{battery}}(\text{camera}) = 1$.

Type Consistency Check: Here we ensure the type modifiers for two targets in the current domain d_{u+1} are consistent with these type modifiers in the past domain $d \in D$. This is because an item can be an aspect in one domain but an entity in another. For example, if the current domain is "Cellphone", borrowing the relation "camera's battery" from domain "Camera" can introduce an error because "camera" is an aspect in domain "Cellphone".

Syntactic pattern "this" is a good indicator for this checking. In the "Cellphone" domain, "its camera" or "the camera" are often mentioned but not "this camera". In the "Camera" domain, "this camera" is often mentioned. The type modifier of "camera" in "Cellphone" is m_A, but in "Camera" it is m_E.

Updating Probabilities in Current Domain d_{u+1}: Edges for RL are in the forms of conditional label distribution $P(L(t_i)|L(t_j))$ and neighborhood weight distribution $w(t_j|t_i)$. We now discuss how to use the KB to estimate them more accurately.

Updating Conditional Label Distribution: Equation (5) tells that conditional label distribution $P(L(t_i)|L(t_j))$ is the weighted sum of relation modifiers' label distributions P_{m_c}, $P_{m_{A|E}}$, and $P_{m_{E|A}}$. These 3 label distributions are pre-defined and given in Table 2. They are not changed. Thus, we update conditional label distribution through updating the three relation modifiers' weights $q(M_{t_j}(t_i))$ with the knowledge in the KB. Recall the three relation modifiers are $M_R = \{m_c, m_{A|E}, m_{E|A}\}$.

After consistency check, there can be multiple relation modifiers between two targets in similar past domains $D^s \subset D$. The number of domains supporting a relation modifier $m \in M_R$ can tell which kind of relation modifiers is common and likely to be correct. For example, given many past domains like "Laptop", "Tablet", "Cellphone", etc., "camera and battery" appears more than "camera's battery", "camera" should be modified by "battery" more with $m_{E|A}$ rather than m_c (likely to be an aspect).

Let $C^{d_{u+1}}_{m,t_j}(t_i)$ be the count that target t_i modified by target t_j on relation m in the current domain d_{u+1} (not in KB). The count $C^{(\text{CL})}$ is for updating the Conditional Label (CL) distributions considering the information in both the current domain d_{u+1} and the KB. It is calculated as:

$$C^{(\text{CL})}_{m,t_j}(t_i) = \begin{cases} C^{d_{u+1}}_{m,t_j}(t_i) & \text{if } C^{d_{u+1}}_{m,t_j}(t_i) > 0 \\ \sum_{d \in D^s} \mathbb{I}^d_{m,t_j}(t_i) & \text{if } \sum_{m \in M_R} C^{d_{u+1}}_{m,t_j}(t_i)) = 0 \end{cases}$$

This equation says that if there is any relation modifier existing between the two targets in the new domain d_{u+1}, we do not borrow edges from the KB; Otherwise, the number of similar past domains supporting the relation modifier m is used. Recall that $\mathbb{I}^d_{m,t_j}(t_i)$ is the result calculated by Equation (7) after label consistency check.

We use count $C^{(\text{CL})}_{m,t_j}(t_i)$ to update $q^{d_{u+1}}(M_{t_j}(t_i))$ using Equation (4) in Section 4.2. Then the conditional label distribution accommodating relation modifiers in the KB, $P^{(\text{LL1})}(L(t_i)|L(t_j))$, is calculated by Equation, (5) using $q^{d_{u+1}}(M_{t_j}(t_i))$. LL1 denotes *Lifelong Learning* 1.

Updating Neighbor Weight Distribution: Equation (6) says that $w(t_j|t_i)$ is the importance of target t_i's neighbor t_j to t_i among all t_i's neighbors. When updating conditional label distribution using the KB, the number of domains can decide which kind of relation modifiers m is more common between the two targets t_i and t_j. But we cannot tell that neighbor t_j is more important than another neighbor $t_{j'}$ to t_i.

For example, given the past domains such as "Laptop", "Tablet", "Cellphone", etc., no matter how many domains believe "camera" is an aspect given "battery" is also an aspect, if the current domain is "All-in-one desktop computer", we should not consider the strong influences from "battery" in the past domains. We should rely more on the weights of "camera"'s neighbors provided by "All-in-one desktop computer". That means "mouse", "keyboard", "screen" etc., should have strong influences on "camera" than "battery" because most All-in-one desktops (e.g. iMac) do not have battery.

We introduce another indicator variable $\mathbb{I}^D_{m,t_j}(t_i) = \bigcup_{d \in D^s} \mathbb{I}^d_{m,t_j}(t_i)$, to indicate whether target t_j modified t_i on relation m in past similar

domains D^s. It only considers the existence of a relation modifier m among domains D^s.

The count $C^{(w)}_{t_j}(t_i)$ for updating the neighbor weight (w) distribution considers both the KB and the current domain d_{u+1}. It is as follows: $C^{(w)}_{t_j}(t_i) = \begin{cases} \sum_{m \in M_R} C^{d_{u+1}}_{m,t_j}(t_i) & \text{if } \sum_{m \in M_R} C^{d_{u+1}}_{m,t_j}(t_i) > 0 \\ \sum_{m \in M_R} \mathbb{I}^{D_u}_{m,t_j}(t_i) & \text{if } \sum_{m \in M_R} C^{d_{u+1}}_{m,t_j}(t_i) = 0 \end{cases}$
This equation tells that if there are relation modifiers existing between the two targets in the new domain d_{u+1}, we count the total times that t_j modifies t_i in the new domain; Otherwise, we count the total kinds of relation modifiers in M_R if a relation modifier $m \in M_R$ existed in past domains. Let $w^{(\text{LL1})}(t_j|t_i)$ be the neighbor weight distribution considering knowledge from the KB and d_{u+1}. It is calculated by Equation (6) using $C^{(w)}_{t_j}(t_i)$.

The initial label distribution $P^{d_{u+1},0}$ is calculated by Equation (3) only using type modifiers found in the new domain d_{u+1}. We use Lifelong-RL-1 to denote the method that employs $P^{(\text{LL1})}(L(t_i)|L(t_j))$, $w^{(\text{LL1})}(t_j|t_i)$ and $P^{d_{u+1},0}$ as inputs for RL.

5.2 Exploiting Target Labels in the KB

Since we have target labels from past domains, we may have a better idea about the initial label probabilities of targets in the current domain d_{u+1}. For example, after labeling domains like "Cellphone", "Laptop", "Tablet," and "E-reader", we may have a good sense that "camera" is likely to be an aspect. To use such knowledge, we need to check if the type modifier of target t in the current domain matches those in past domains and only keep those domains that have such a matching type modifier.

Let $D^s \subset D$ be the past domains consistent with target t's type modifier in the current domain d_{u+1}. Let $C^{D^s}(L(t))$ be the number of domains in D^s that target t is labeled as $L(t)$. Let λ be the ratio that controls how much we trust knowledge from the KB. Then the initial label probability distribution $P^{d_{u+1},0}$ calculated by Equation (3) only using type modifier found in d_{u+1} is replaced by :

$$P^{(\text{LL2}),0}(L(t)) = \frac{|D| \times P^{d_{u+1},0}(L(t)) + \lambda C^{D^s}(L(t))}{|D| + \lambda |D|} \tag{8}$$

Similarly, let $D^s \subset D$ be the past domains consistent with both targets t_i's and t_j's type modifiers in d_{u+1}. Let $C^{D^s}(L(t_i), L(t_j))$ be the number of domains in D^s that t_i and t_j are labeled as $L(t_i)$ and $L(t_j)$ respectively. The conditional label probability distribution accommodating relation modifiers in the KB, $P^{(\text{LL1})}(L(t_i)|L(t_j))$, is further updated to $P^{(\text{LL2})}(L(t_i)|L(t_j))$ by exploiting the target labels in KB (LL2 denotes *Lifelong Learning* 2):

$$P^{(\text{LL2})}(L(t_i)|L(t_j)) = \frac{|D| \times P^{(\text{LL1})}(L(t_i)|L(t_j)) + \lambda C^{D^s}(L(t_i), L(t_j))}{|D| + \lambda |D|} \tag{9}$$

For example, given "this camera", "battery" in the current domain, we are more likely to consider domains (e.g. "Film Camera", "DSLR", but not "Cellphone") that have entity modifiers on "camera" and aspect modifiers on "battery". Then we count the number of those domains that label "camera" as entity and "battery" as aspect: $C^{D^s}(L(\text{camera}) = \text{entity}, L(\text{battery}) = \text{aspect})$. Similarly, we count domains having other types of target labels on "camera" and "battery". These counts form an updated conditional label distribution that estimates "camera" as an entity and "battery" as an aspect.

Note that $|D - D^s|$, the number of past domains not consistent with targets' type modifiers, is added to $C^{D^s}(L(t_i) = \text{NIL})$ and $C^{D^s}(L(t_i) = \text{NIL}, L(t_j))$ for Equations (8) and (9) respectively to make the sum over $L(t_i)$ equal to 1. We use Lifelong-RL to denote this method which uses $P^{(\text{LL2}),0}(L(t))$, $P^{(\text{LL2})}(L(t_i)|L(t_j))$ and $w^{(\text{LL1})}(t_j|t_i)$ as input for RL.

6 Experiments

We now evaluate the proposed method and compare with baselines. We use the DP method for target extraction (Qiu et al., 2011). This method uses dependency relations between opinion words and targets to extract targets using seed opinion words. Since our paper does not focus on extraction, interested readers can refer to (Qiu et al., 2011) for details.

6.1 Experiment Settings

Evaluation Datasets: We use two sets of datasets. The first set consists of eight (8) annotated review datasets. We use each of them as the new domain data in LML to compute precision, recall, F1 scores. Five of them are from (Hu and Liu, 2004), and the remaining three are from (Liu et al., 2016). They have been used for target extraction, and thus have annotated targets, but no annotation on whether a

Dataset	Product Type	# of Sentence	# of entity	# of aspect
D1	Computer	531	50	151
D2	Wireless Router	879	97	186
D3	Speaker	689	64	218
D4	DVD Player	740	50	159
D5	Digital Camera	597	70	239
D6	MP3 Player	1716	60	370
D7	Digital Camera	346	28	151
D8	Cell Phone	546	36	188

Table 1: Annotation details of the benchmark datasets.

Distribution	$L(t) =$ entity	$L(t) =$ aspect	$L(t) =$ NIL
$\mathbf{P_{m_E}}$	0.45	0.25	0.3
$\mathbf{P_{m_A}}$	0.3	0.4	0.3

$\mathbf{P_{m_c}}$	$L(t_j) =$ entity	$L(t_j) =$ aspect	$L(t_j) =$ NIL
$L(t_i) =$ entity	0.8	0.0	0.33
$L(t_i) =$ aspect	0.0	0.8	0.33
$L(t_i) =$ NIL	0.2	0.2	0.33
$\mathbf{P_{m_{E\|A}}}$	$L(t_j) =$ entity	$L(t_j) =$ aspect	$L(t_j) =$ NIL
$L(t_i) =$ entity	0.33	0.8	0.33
$L(t_i) =$ aspect	0.33	0.0	0.33
$L(t_i) =$ NIL	0.33	0.2	0.33
$\mathbf{P_{m_{A\|E}}}$	$L(t_j) =$ entity	$L(t_j) =$ aspect	$L(t_j) =$ NIL
$L(t_i) =$ entity	0.0	0.33	0.33
$L(t_i) =$ aspect	0.8	0.33	0.33
$L(t_i) =$ NIL	0.2	0.33	0.33

Table 2: Label Distribution for P_E and P_A and Conditional Label Distribution for P_{m_c}, $P_{m_{A|E}}$ and $P_{m_{E|A}}$

target is an entity or aspect. We made this annotation, which is straightforward. We used two annotators to annotate the datasets. The Cohen's kappa is 0.84. Through discussion, the annotators got complete agreement. Details of the datasets are listed in Table 1. Each cell is the number of distinct terms. These datasets are not very large but they are realistic because many products do not have a large number of reviews.

The second set consists of unlabeled review datasets from 100 diverse products or domains (Chen and Liu 2014). Each domain has 1000 reviews. They are treated as past domain data in LML since they are not annotated and thus cannot be used for computing evaluation measures.

Evaluating Measures: We mainly use precision $\mathcal{P}$, recall $\mathcal{R}$, and F_1-score $\mathcal{F}_1$ as evaluation measures. We take multiple occurrences of the same target as one count, and only evaluate entities and aspects. We will also give the accuracy results.

Compared Methods: We compare the following methods, including our proposed method, *Lifelong-RL*.

NER+TM: NER is Named Entity Recognition. We can regard the extracted terms from a NER system as entities and the rest of the targets as aspects. However, a NER system cannot identify entities such as "this car" from "this car is great." Its result is rather poor. But our type modifier (TM) does that, i.e., if an opinion target appears after "this" or "these" in at least two sentences, TM labels the target as an entity; otherwise an aspect. However, TM cannot extract named entities. Its result is also rather poor. We thus combine the two methods to give NER+TM as they complement each other very well. To make NER more powerful, we use two NER systems: Stanford-NER [1](Manning et al., 2014) and UIUC-NER[2] (Ratinov and Roth, 2009). NER+TM treats the extracted entities by the three systems as entities and the rest of the targets as aspects.

NER+TM+DICT: We run NER+TM on the 100 datasets for LML to get a list of entities, which we call the $dictionary$ (DICT). For a new task, if any target word is in the list, it is treated as an entity; otherwise an aspect.

RL: This is the base method described in Section 3. It performs relaxation labeling (RL) without the help of LML.

Lifelong-RL-1: This performs LML with RL but the current task only uses the relations in the KB from previous tasks (Section 5.1).

Lifelong-RL: This is our proposed final method. It improves Lifelong-RL-1 by further incorporating target labels in the KB from previous tasks (Section 5.2).

Parameter Settings: RL has 2 initial label distributions P_{m_E} and P_{m_A} and 3 conditional label distributions P_{m_c}, $P_{m_{E|A}}$ and $P_{m_{A|E}}$. Like other belief propagation algorithms, these probabilities need to be set empirically, as shown in Table 2. The parameter α is set to 1. Our LML method has one parameter λ for Lifelong-RL. We set it to 0.1.

6.2 Results Analysis

Table 3 shows the test results of all systems in precision, recall and F_1-score except NER+TM+DICT. NER+TM+DICT is not included due to space limitations and because it performed very poorly. The

[1]http://nlp.stanford.edu/software/CRF-NER.shtml

[2]https://cogcomp.cs.illinois.edu/page/software_view/NETagger

Dataset	Entity												Aspect											
	NER+TM			RL			Lifelong-RL-1			Lifelong-RL			NER+TM			RL			Lifelong-RL-1			Lifelong-RL		
	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$	$\mathcal{P}$	$\mathcal{R}$	$\mathcal{F}_1$
D1	56.3	88	68.7	80	56	65.9	76.1	70	72.9	83.1	66	73.6	71.0	74.8	72.9	72.6	74.2	73.4	75.3	71.5	73.4	74.2	72.8	73.5
D2	64.8	75.3	69.7	71.6	42.3	53.2	81.1	62.9	70.9	85.5	78.4	81.8	61.9	90.3	73.5	61.4	85	71.3	67.2	92.5	77.9	70.4	90.9	79.3
D3	56.8	68.6	62.2	63.4	37.5	47.1	79.8	62.5	70.1	76.3	64.1	69.6	76.3	81.7	78.9	76.6	77.5	77.1	73.7	84.4	78.7	73.5	82.6	77.8
D4	76.7	42	54.3	69.3	42	52.3	77.9	70	73.7	78.6	70	74	68.8	71.7	70.2	68.3	70.4	69.3	70.4	65.4	67.8	70.6	66	68.2
D5	62.7	54.3	58.2	62.1	61.4	61.8	78.5	94.3	85.7	86.4	91.4	88.9	85.6	81.6	83.5	85.5	77.8	81.5	87	81.2	84	87.7	82	84.8
D6	69.9	38.3	49.5	67	56.7	61.4	74.7	75	74.8	77.4	73.3	75.3	75.4	83	79	76.2	81.1	78.6	78.8	85.9	82.2	78.9	86.2	82.4
D7	95	64.28	76.7	95.2	67.9	79.2	93.8	92.8	93.3	94.7	92.9	93.8	87.5	86.1	86.8	87.9	86.8	87.3	89.1	88.1	88.6	90.7	88.7	89.7
D8	65.9	41.7	51.1	65.5	72.2	68.7	72.3	83.3	77.4	79.4	86.1	82.6	76.1	81.9	78.9	77.8	80.9	79.3	81.4	89.4	85.2	81.9	89.9	85.7
Average	68.5	59.1	61.3	71.8	54.5	61.2	79.3	76.4	77.4	82.7	77.8	79.9	75.3	81.4	78	75.8	79.2	77.2	77.9	82.3	79.7	78.5	82.4	80.2

Table 3: Comparative results on Entity and Aspect in precision, recall and F_1 score: NER+TM+DICT's results are very poor and not included (see Section 6.2) for the average results.

reason is that a target can be an entity in one domain but an aspect in another. Its average F_1-score for entity is only 49.2, and for aspect is only 50.2.

Entity Results Comparison: We observe from the table that although NER+TM combines NER and TM, its result for entities is still rather poor. We notice that phrases like "this price" causes low precision. Since it does not use many other relations and NER does not recognize many named entities that are written in lower case letters (e.g., "apple is good"), its recall is also low.

RL has a higher precision as it considers relation modifiers. However, its recall is low because it lacks information in its graph, which causes RL to make many wrong decisions. Lifelong-RL-1 introduces relation modifiers in KB from past domains into the current task. Both precision and recall increase markedly.

Lifelong-RL improves Lifelong-RL-1 further by considering target labels of past domains. Their counts improve the initial label probability distributions and conditional label probability distributions. For example, "this price" may appear in some domains but "price"'s target label is mostly aspect. We consider their counts in initial label distributions and thus rectify the initial distribution of "price". This makes "price" easier to be classified as aspect and thus improves the precision for entity.

Aspect Results Comparison: For aspects, the trend is the same but the improvements are not as dramatic as for entity. This is because the distribution of entity and aspect in the data is highly skewed. There are many more aspects than entities as we can see from the Table 1. When an entity term is wrongly classified as an aspect, it has much less impact on the aspect result than on the entity result.

Accuracy Results Comparison: Table 4 gives the classification accuracy results considering all three classes. We can see the similar trend. NER+TM+DICT's average accuracy is only 45.89 and is not included in the table.

Dataset	NER+TM	RL	Lifelong-RL-1	Lifelong-RL
D1	64.93	74.29	75.51	76.34
D2	62.94	63.53	69.8	73.82
D3	70.04	73.74	74.83	74.1
D4	70.81	68.57	73.33	73.63
D5	82.07	81.46	85.22	87.5
D6	74.83	75.06	78	78.63
D7	88.18	88.63	89.68	91.3
D8	74.54	75.43	79.57	81.4
Average	73.55	75.07	78.24	79.59

Table 4: Results in accuracy: NER+TM+DICT's results are again very poor and thus not included.

7 Conclusion

This paper studied the problem of classifying opinion targets into entities and aspects. To the best of our knowledge, this problem has not been attempted in the unsupervised opinion target extraction setting. But this is an important problem because without separating or classifying them one will not know whether an opinion is about an entity as a whole or about a specific aspect of an entity. This paper proposed a novel method based on relaxation labeling and the paradigm of lifelong machine learning to solve the problem. Experimental results showed the effectiveness of the proposed method.

Acknowledgments

This work was partially supported by National Science Foundation (NSF) grants IIS-1407927 and IIS-1650900, and NCI grant R01CA192240. The content of the paper is solely the responsibility of the authors and does not necessarily represent the official views of the NSF or NCI.

References

Samuel Brody and Noemie Elhadad. 2010. An unsupervised aspect-sentiment model for online reviews. In *NAACL '10*, pages 804–812.

Rich Caruana. 1997. Multitask learning. *Machine learning*, 28(1):41–75.

Zhiyuan Chen and Bing Liu. 2014a. Mining topics in documents: Standing on the shoulders of big data. In *KDD '14*, pages 1116–1125.

Zhiyuan Chen and Bing Liu. 2014b. Topic modeling using topics from many domains, lifelong learning and big data. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, pages 703–711.

Zhiyuan Chen, Nianzu Ma, and Bing Liu. 2015. Lifelong learning for sentiment classification. *Volume 2: Short Papers*, pages 750–756.

Yejin Choi and Claire Cardie. 2010. Hierarchical sequential learning for extracting opinions and their attributes. In *ACL '10*, pages 269–274.

Lei Fang and Minlie Huang. 2012. Fine granular aspect analysis using latent structural models. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2*, pages 333–337.

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.

Robert A Hummel and Steven W Zucker. 1983. On the foundations of relaxation labeling processes. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, (3):267–287.

Niklas Jakob and Iryna Gurevych. 2010. Extracting opinion targets in a single- and cross-domain setting with conditional random fields. In *EMNLP '10*, pages 1035–1045.

Dan Klein and Christopher D Manning. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics-Volume 1*, pages 423–430.

Lun-Wei Ku, Yu-Ting Liang, and Hsin-Hsi Chen. 2006. Opinion extraction, summarization and tracking in news and blog corpora. In *AAAI spring symposium: Computational approaches to analyzing weblogs*, volume 100107.

Fangtao Li, Minlie Huang, and Xiaoyan Zhu. 2010. Sentiment analysis with global topics and local dependency. In *AAAI '10*, pages 1371–1376.

Chenghua Lin and Yulan He. 2009. Joint sentiment/topic model for sentiment analysis. In *Proceedings of the 18th ACM conference on Information and knowledge management*, pages 375–384.

Kang Liu, Liheng Xu, and Jun Zhao. 2013. Syntactic patterns versus word alignment: Extracting opinion targets from online reviews. In *ACL (1)*, pages 1754–1763.

Qian Liu, Bing Liu, Yuanlin Zhang, DooSoon Kim, and Zhiqiang Gao. 2016. Improving opinion aspect extraction using semantic similarity and aspect associations. In *AAAI*.

Bing Liu. 2012. Sentiment analysis and opinion mining. *Synthesis lectures on human language technologies*, 5(1):1–167.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Association for Computational Linguistics (ACL) System Demonstrations*, pages 55–60.

Qiaozhu Mei, Xu Ling, Matthew Wondra, Hang Su, and ChengXiang Zhai. 2007. Topic sentiment mixture: Modeling facets and opinions in weblogs. In *WWW '07*, pages 171–180.

Margaret Mitchell, Jacqui Aguilar, Theresa Wilson, and Benjamin Van Durme. 2013. Open domain targeted sentiment. In *ACL '13*, pages 1643–1654.

Arjun Mukherjee and Bing Liu. 2012. Aspect extraction through semi-supervised modeling. In *ACL '12*, volume 1, pages 339–348.

Sinno Jialin Pan and Qiang Yang. 2010. A survey on transfer learning. *Knowledge and Data Engineering, IEEE Transactions on*, 22(10):1345–1359.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and trends in information retrieval*, 2(1-2):1–135.

Ana-Maria Popescu and Orena Etzioni. 2007. Extracting product features and opinions from reviews. In *Natural language processing and text mining*, pages 9–28. Springer.

Soujanya Poria, Erik Cambria, Lun-Wei Ku, Chen Gui, and Alexander Gelbukh. 2014. A rule-based approach to aspect extraction from product reviews. *SocialNLP 2014*, page 28.

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.

L. Ratinov and D. Roth. 2009. Design challenges and misconceptions in named entity recognition. In *CoNLL*, 6.

Paul Ruvolo and Eric Eaton. 2013. Active task selection for lifelong machine learning. In *AAAI*.

Daniel L Silver, Qiang Yang, and Lianghao Li. 2013. Lifelong machine learning systems: Beyond learning algorithms. In *AAAI Spring Symposium: Lifelong Machine Learning*, pages 49–55.

Sebastian Thrun. 1998. Lifelong learning algorithms. In *Learning to learn*, pages 181–209. Springer.

Bo Wang and Houfeng Wang. 2008. Bootstrapping both product features and opinion words from chinese customer reviews with cross-inducing. In *IJCNLP '08*, pages 289–295.

Hongning Wang, Yue Lu, and Chengxiang Zhai. 2010. Latent aspect rating analysis on review text data: A rating regression approach. In *KDD '10*, pages 783–792.

Yuanbin Wu, Qi Zhang, Xuanjing Huang, and Lide Wu. 2009. Phrase dependency parsing for opinion mining. In *EMNLP '09*, pages 1533–1541.

Lei Zhang, Bing Liu, Suk Hwan Lim, and Eamonn O'Brien-Strain. 2010. Extracting and ranking product features in opinion documents. In *COLING '10: Posters*, pages 1462–1470.

Xinjie Zhou, Xiaojun Wan, and Jianguo Xiao. 2013. Collective opinion target extraction in chinese microblogs. In *EMNLP*, volume 13, pages 1840–1850.

Li Zhuang, Feng Jing, and Xiao-Yan Zhu. 2006. Movie review mining and summarization. In *CIKM '06*, pages 43–50.