Recursive Neural Conditional Random Fields for Aspect-based Sentiment Analysis

Wenya Wang[†‡] **Sinno Jialin Pan**[†] **Daniel Dahlmeier**[‡] **Xiaokui Xiao**[†]
[†]Nanyang Technological University, Singapore
[‡]SAP Innovation Center Singapore
[†]{wa0001ya, sinnopan, xkxiao}@ntu.edu.sg, [‡]{d.dahlmeier}@sap.com

Abstract

In aspect-based sentiment analysis, extracting aspect terms along with the opinions being expressed from user-generated content is one of the most important subtasks. Previous studies have shown that exploiting connections between aspect and opinion terms is promising for this task. In this paper, we propose a novel joint model that integrates recursive neural networks and conditional random fields into a unified framework for explicit aspect and opinion terms co-extraction. The proposed model learns high-level discriminative features and double propagates information between aspect and opinion terms, simultaneously. Moreover, it is flexible to incorporate hand-crafted features into the proposed model to further boost its information extraction performance. Experimental results on the dataset from SemEval Challenge 2014 task 4 show the superiority of our proposed model over several baseline methods as well as the winning systems of the challenge.

1 Introduction

Aspect-based sentiment analysis (Pang and Lee, 2008; Liu, 2011) aims to extract important information, e.g., opinion targets, opinion expressions, target categories, and opinion polarities, from user-generated content, such as microblogs, reviews, etc. This task was first studied by Hu and Liu (2004a; 2004b), followed by Popescu and Etzioni (2005), Zhuang et al. (2006), Zhang et al. (2010), Qiu et al. (2011), Li et al. (2010). In aspect-based sentiment analysis, one of the goals is to extract explicit aspects of an entity from text, along with the opinions being expressed. For example, in a restaurant review *"I have to say they have one of the fastest delivery times in the city."*, the aspect term is *delivery times*, and the opinion term is *fastest*.

Among previous work, one of the approaches is to accumulate aspect and opinion terms from a seed collection without label information, by utilizing syntactic rules or modification relations between them (Qiu et al., 2011; Liu et al., 2013b). In the above example, if we know *fastest* is an opinion word, then *delivery times* is probably inferred to be an aspect because *fastest* is its modifier. However, this approach largely relies on hand-coded rules and is restricted to certain Part-of-Speech (POS) tags, e.g., opinion words are restricted to be adjectives. Another approach focuses on feature engineering based on predefined lexicons, syntactic analysis, etc. (Jin and Ho, 2009; Li et al., 2010). A sequence labeling classifier is then built to extract aspect and opinion terms. This approach requires extensive efforts for designing hand-crafted features and only combines features linearly for classification which ignores higher order interactions.

To overcome the limitations of existing methods, we propose a novel model, named Recursive Neural Conditional Random Fields (RNCRF). Specifically, RNCRF consists of two main components. The first component is a recursive neural network (RNN)[1] (Socher et al., 2010) based on a dependency tree of each sentence. The goal is to learn a high-level feature representation for each word in the context of each sentence and make the representation learning for aspect and opinion terms interactive through the underlying dependency structure among them. The output of the RNN is then fed into a Conditional Random Field (CRF) (Lafferty et al., 2001) to learn a discriminative mapping from high-

[1] Note that in this paper, RNN stands for recursive neural network instead of recurrent neural network.

Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, pages 616–626,
Austin, Texas, November 1-5, 2016. ©2016 Association for Computational Linguistics

level features to labels, i.e., *aspects*, *opinions*, or *others*, so that context information can be well captured. Our main contributions are to use RNN for encoding aspect-opinion relations in high-level representation learning and to present a joint optimization approach based on maximum likelihood and backpropagation to learn the RNN and CRF components simultaneously. In this way, the label information of aspect and opinion terms can be dually propagated from parameter learning in CRF to representation learning in RNN. We conduct expensive experiments on the dataset from SemEval challenge 2014 task 4 (subtask 1) (Pontiki et al., 2014) to verify the superiority of RNCRF over several baseline methods as well as the winning systems of the challenge.

2 Related Work

2.1 Aspects and Opinions Co-Extraction

Hu et al. (2004a) proposed to extract product aspects through association mining, and opinion terms by augmenting a seed opinion set using synonyms and antonyms in WordNet. In follow-up work, syntactic relations are further exploited for aspect/opinion extraction (Popescu and Etzioni, 2005; Wu et al., 2009; Qiu et al., 2011). For example, Qiu et al. (2011) used syntactic relations to double propagate and augment the sets of aspects and opinions. Although the above models are unsupervised, they heavily depend on predefined rules for extraction and are restricted to specific types of POS tags for product aspects and opinions. Jin et al. (2009), Li et al. (2010), Jakob et al. (2010) and Ma et al. (2010) modeled the extraction problem as a sequence tagging problem and proposed to use HMMs or CRFs to solve it. These methods rely on rich hand-crafted features and do not consider interactions between aspect and opinion terms explicitly. Another direction is to use word alignment model to capture opinion relations among a sentence (Liu et al., 2012; Liu et al., 2013a). This method requires sufficient data for modeling desired relations.

Besides explicit aspects and opinions extraction, there are also other lines of research related to aspect-based sentiment analysis, including aspect classification (Lakkaraju et al., 2014; McAuley et al., 2012), aspect rating (Titov and McDonald, 2008; Wang et al., 2011; Wang and Ester, 2014), domain-specific and target-dependent sentiment classification (Lu et al., 2011; Ofek et al., 2016; Dong et al., 2014; Tang et al., 2015).

2.2 Deep Learning for Sentiment Analysis

Recent studies have shown that deep learning models can automatically learn the inherent semantic and syntactic information from data and thus achieve better performance for sentiment analysis (Socher et al., 2011b; Socher et al., 2012; Socher et al., 2013; Glorot et al., 2011; Kalchbrenner et al., 2014; Kim, 2014; Le and Mikolov, 2014). These methods generally belong to sentence-level or phrase/word-level sentiment polarity predictions. Regarding aspect-based sentiment analysis, Irsoy et al. (2014) applied deep recurrent neural networks for opinion expression extraction. Dong et al. (2014) proposed an adaptive recurrent neural network for target-dependent sentiment classification, where targets or aspects are given as input. Tang et al. (2015) used Long Short-Term Memory (LSTM) (Hochreiter and Schmidhuber, 1997) for the same task. Nevertheless, there is little work in aspects and opinions co-extraction using deep learning models.

To the best of our knowledge, the work of Liu et al. (2015) and Yin et al. (2016) are the most related to ours. Liu et al. (2015) proposed to combine recurrent neural network and word embeddings to extract explicit aspects. However, the proposed model simply uses recurrent neural network on top of word embeddings, and thus its performance heavily depends on the quality of word embeddings. In addition, it fails to explicitly model dependency relations or compositionalities within certain syntactic structure in a sentence. Recently, Yin et al. (2016) proposed an unsupervised learning method to improve word embeddings using dependency path embeddings. A CRF is then trained with the embeddings independently in the pipeline.

Different from (Yin et al., 2016), our model does not focus on developing a new unsupervised word embedding methods, but encodes the information of dependency paths into RNN for constructing syntactically meaningful and discriminative hidden representations with labels. Moreover, we integrate RNN and CRF into a unified framework and develop a joint optimization approach, instead of training word embeddings and a CRF separately as in (Yin et al., 2016). Note that Weiss et al. (2015) proposed to combine deep learning and structured learning for language parsing which can be learned by structured perceptron. However, they also separate neural network training with structured prediction.

2.3 Recursive Neural Networks

Among deep learning methods, RNN has shown promising results on various NLP tasks, such as learning phrase representations (Socher et al., 2010), sentence-level sentiment analysis (Socher et al., 2013), language parsing (Socher et al., 2011a), and question answering (Iyyer et al., 2014). The tree structures used for RNNs include constituency tree and dependency tree. In a constituency tree, all the words lie at leaf nodes, each internal node represents a phrase or a constituent of a sentence, and the root node represents the entire sentence (Socher et al., 2010; Socher et al., 2012; Socher et al., 2013). In a dependency tree, each node including terminal and nonterminal nodes, represents a word, with dependency connections to other nodes (Socher et al., 2014; Iyyer et al., 2014). The resultant model is known as dependency-tree RNN (DT-RNN). An advantage of using dependency tree over the other is the ability to extract word-level representations considering syntactic relations and semantic robustness. Therefore, we adopt DT-RNN in this work.

3 Problem Statement

Suppose that we are given a training set of customer reviews in a specific domain, denoted by $S=\{s_1, ..., s_N\}$, where N is the number of review sentences. For any $s_i \in S$, there may exist a set of aspect terms $A_i = \{a_{i1}, ..., a_{il}\}$, where each $a_{ij} \in A_i$ can be a single word or a sequence of words expressing explicitly some aspect of an entity, and a set of opinion terms $O_i = \{o_{i1}, ..., o_{im}\}$, where each o_{ir} can be a single word or a sequence of words expressing the subjective sentiment of the comment holder. The task is to learn a classifier to extract the set of aspect terms A_i and the set of opinion terms O_i from each review sentence $s_i \in S$.

This task can be formulated as a sequence tagging problem by using the BIO encoding scheme. Specifically, each review sentence s_i is composed of a sequence of words $s_i = \{w_{i1}, ..., w_{in_i}\}$. Each word $w_{ip} \in s_i$ is labeled as one out of the following 5 classes: BA (beginning of aspect), IA (inside of aspect), BO (beginning of opinion), IO (inside of opinion), and O (others). Let $\mathbf{L} = \{\mathrm{BA}, \mathrm{IA}, \mathrm{BO}, \mathrm{IO}, \mathrm{O}\}$. We are also given a test set of review sentences denoted by $S' = \{s'_1, ..., s'_{N'}\}$, where N' is the number of test reviews. For each test review $s'_i \in S'$, our objective is to predict the class label $y'_{iq} \in \mathbf{L}$ for each word w'_{iq}. Note that a sequence of predictions with BA at the beginning followed by IA are indication of one aspect, which is similar for opinion terms.[2]

4 Recursive Neural CRFs

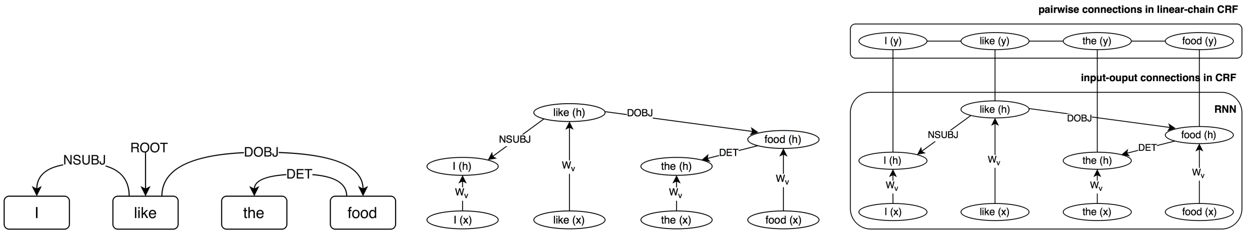
As described in Section 1, RNCRF consists of two main components: 1) a DT-RNN to learn a high-level representation for each word in a sentence, and 2) a CRF to take the learned representation as input to capture context around each word for explicit aspect and opinion terms extraction. Next, we present these two components in details.

4.1 Dependency-Tree RNNs

We begin by associating each word w in our vocabulary with a feature vector $x \in \mathbb{R}^d$, which corresponds to a column of a word embedding matrix $W_e \in \mathbb{R}^{d \times v}$, where v is the size of the vocabulary. For each sentence, we build a DT-RNN based on the corresponding dependency parse tree with word embeddings as initialization. An example of the dependency parse tree is shown in Figure 1(a), where each edge starts from the parent and points to its dependent with a syntactic relation.

In a DT-RNN, each node n, including leaf nodes, internal nodes and the root node, in a specific sentence is associated with a word w, an input feature vector x_w, and a hidden vector $h_n \in \mathbb{R}^d$ of the same dimension as x_w. Each dependency relation r is associated with a separate matrix $W_r \in \mathbb{R}^{d \times d}$. In addition, a common transformation matrix $W_v \in \mathbb{R}^{d \times d}$ is introduced to map the word embedding x_w at node n to its corresponding hidden vector h_n.

[2] In this work we focus on extraction of aspect and opinion terms, not polarity predictions on opinion terms. Polarity prediction can be done by either post-processing on the extracted opinion terms or redefining the BIO labels by encoding the polarity information.



(a) Example of a dependency tree. (b) Example of a DT-RNN tree structure. (c) Example of a RNCRF structure.

Figure 1: Examples of dependency tree, DT-RNN structure and RNCRF structure for a review sentence.

Along with a particular dependency tree, a hidden vector h_n is computed from its own word embedding x_w at node n with the transformation matrix W_v and its children's hidden vectors $h_{\text{child(n)}}$ with the corresponding relation matrices $\{W_r\}$'s. For instance, given the parse tree shown in Figure 1(a), we first compute the leaf nodes associated with the words *I* and *the* using W_v as follows,

$$\begin{aligned} h_{\text{I}} &= f(W_v \cdot x_{\text{I}} + b), \\ h_{\text{the}} &= f(W_v \cdot x_{\text{the}} + b), \end{aligned}$$

where f is a nonlinear activation function and b is a bias term. In this paper, we adopt $tanh(\cdot)$ as the activation function. Once the hidden vectors of all the leaf nodes are generated, we can recursively generate hidden vectors for interior nodes using the corresponding relation matrix W_r and the common transformation matrix W_v as follows,

$$\begin{aligned} h_{\text{food}} &= f(W_v \cdot x_{\text{food}} + W_{\text{DET}} \cdot h_{\text{the}} + b), \\ h_{\text{like}} &= f(W_v \cdot x_{\text{like}} + W_{\text{DOBJ}} \cdot h_{\text{food}} \\ &\quad + W_{\text{NSUBJ}} \cdot h_{\text{I}} + b). \end{aligned}$$

The resulting DT-RNN is shown in Figure 1(b). In general, a hidden vector for any node n associated with a word vector x_w can be computed as follows,

$$h_n = f\left(W_v \cdot x_w + b + \sum_{k \in \mathcal{K}_n} W_{r_{nk}} \cdot h_k\right), \quad (1)$$

where $\mathcal{K}_n$ denotes the set of children of node n, r_{nk} denotes the dependency relation between node n and its child node k, and h_k is the hidden vector of the child node k. The parameters of DT-RNN, $\Theta_{\text{RNN}} = \{W_v, W_r, W_e, b\}$, are learned during training.

4.2 Integration with CRFs

CRFs are a discriminative graphical model for structured prediction. In RNCRF, we feed the output of DT-RNN, i.e., the hidden representation of each word in a sentence, to a CRF. Updates of parameters for RNCRF are carried out successively from the top to bottom, by propagating errors through CRF to the hidden layers of RNN (including word embeddings) using backpropagation through structure (BPTS) (Goller and Küchler, 1996).

Formally, for each sentence s_i, we denote the input for CRF by $\mathbf{h}_i$, which is generated by DT-RNN. Here $\mathbf{h}_i$ is a matrix with columns of hidden vectors $\{h_{i1}, ..., h_{in_i}\}$ to represent a sequence of words $\{w_{i1}, ..., w_{in_i}\}$ in a sentence s_i. The model computes a structured output $\mathbf{y}_i = \{y_{i1}, ..., y_{in_i}\} \in \mathbf{Y}$, where $\mathbf{Y}$ is a set of possible combinations of labels in label set $\mathbf{L}$. The entire structure can be represented by an undirected graph $G = (V, E)$ with cliques $c \in C$. In this paper, we employed linear-chain CRF, which has two different cliques: unary cliques (U) representing input-output connection, and pairwise cliques (P) representing adjacent output connections, as shown in Figure 1(c). During inference, the model aims to output $\hat{\mathbf{y}}$ with the maximum conditional probability $p(\mathbf{y}|\mathbf{h})$. (We drop the subscript i here for simplicity.) The probability is computed from potential outputs of the cliques:

$$p(\mathbf{y}|\mathbf{h}) = \frac{1}{Z(\mathbf{h})} \prod_{c \in C} \psi_c(\mathbf{h}, \mathbf{y}_c), \quad (2)$$

where $Z(\mathbf{h})$ is the normalization term, and $\psi_c(\mathbf{h}, \mathbf{y}_c)$ is the potential of clique c, computed as $\psi_c(\mathbf{h}, \mathbf{y}_c) = \exp \langle W_c, F(\mathbf{h}, \mathbf{y}_c) \rangle$, where the RHS is the exponential of a linear combination of feature vector $F(\mathbf{h}, \mathbf{y}_c)$ for clique c, and the weight vector W_c is tied for unary and pairwise cliques. We

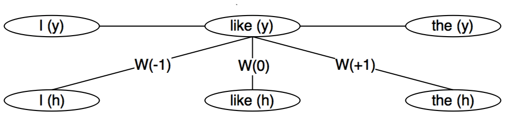


Figure 2: An example for computing input-ouput potential for the second position *like*.

also incorporate a context window of size $2T+1$ when computing unary potentials (T is a hyper-parameter). Thus, the potential of unary clique at node k can be written as

$$\psi_U(\mathbf{h}, y_k) = \exp\Bigg((W_0)_{y_k} \cdot h_k + \sum_{t=1}^{T} (W_{-t})_{y_k} \cdot h_{k-t} + \sum_{t=1}^{T} (W_{+t})_{y_k} \cdot h_{k+t} \Bigg), \tag{3}$$

where W_0, W_{+t} and W_{-t} are weight matrices of the CRF for the current position, the t-th position to the right, and the t-th position to the left within context window, respectively. The subscript y_k indicates the corresponding row in the weight matrix.

For instance, Figure 2 shows an example of window size 3. At the second position, the input features for *like* are composed of the hidden vectors at position 1 (h_{I}), position 2 (h_{like}) and position 3 (h_{the}). Therefore, the conditional distribution for the entire sequence $\mathbf{y}$ in Figure 1(c) can be calculated as

$$p(\mathbf{y}|\mathbf{h}) = \frac{1}{Z(\mathbf{h})} \exp\Bigg(\sum_{k=1}^{4} (W_0)_{y_k} \cdot h_k + \sum_{k=2}^{4} (W_{-1})_{y_k} \cdot h_{k-1} + \sum_{k=1}^{3} (W_{+1})_{y_k} \cdot h_{k+1} + \sum_{k=1}^{3} V_{y_k, y_{k+1}} \Bigg),$$

where the first three terms in the exponential of the RHS consider unary clique while the last term considers the pairwise clique with matrix V representing pairwise state transition score. For simplicity in description on parameter updates, we denote the log-potential for clique $c \in \{U, P\}$ by $g_c(\mathbf{h}, \mathbf{y}_c) = \langle W_c, F(\mathbf{h}, \mathbf{y}_c) \rangle$.

4.3 Joint Training for RNCRF

Through the objective of maximum likelihood, updates for parameters of RNCRF are first conducted on the parameters of the CRF (unary weight matrices $\Theta_U = \{W_0, W_{+t}, W_{-t}\}$ and pairwise weight matrix V) by applying chain rule to log-potential updates. Below is the gradient for Θ_U (updates for V are similar through the log-potential of pairwise clique $g_P(y'_k, y'_{k+1})$):

$$\triangle\Theta_U = \frac{\partial - \log p(\mathbf{y}|\mathbf{h})}{\partial g_U(\mathbf{h}, y'_k)} \cdot \frac{\partial g_U(\mathbf{h}, y'_k)}{\partial \Theta_U}, \tag{4}$$

where

$$\frac{\partial - \log p(\mathbf{y}|\mathbf{h})}{\partial g_U(\mathbf{h}, y'_k)} = -(1_{y_k = y'_k} - p(y'_k|\mathbf{h})), \tag{5}$$

and y'_k represents possible label configuration of node k. The hidden representations of each word and the parameters of DT-RNN are updated subsequently by applying chain rule with (5) through BPTS as follows,

$$\triangle h_{\text{root}} = \frac{\partial - \log p(\mathbf{y}|\mathbf{h})}{\partial g_U(\mathbf{h}, y'_{\text{root}})} \cdot \frac{\partial g_U(\mathbf{h}, y'_{\text{root}})}{\partial h_{\text{root}}}, \tag{6}$$

$$\triangle h_{k \neq \text{root}} = \frac{\partial - \log p(\mathbf{y}|\mathbf{h})}{\partial g_U(\mathbf{h}, y'_k)} \cdot \frac{\partial g_U(\mathbf{h}, y'_k)}{\partial h_k} + \triangle h_{\text{par(k)}} \cdot \frac{\partial h_{\text{par}(k)}}{\partial h_k}, \tag{7}$$

$$\triangle\Theta_{\text{RNN}} = \sum_{k=1}^{K} \frac{\partial - \log p(\mathbf{y}|\mathbf{h})}{\partial h_k} \cdot \frac{\partial h_k}{\partial \Theta_{\text{RNN}}}, \tag{8}$$

where h_{root} represents the hidden vector of the word pointed by ROOT in the corresponding DT-RNN. Since this word is the topmost node in the tree, it only inherits error from the CRF output. In (7), $h_{\text{par}(k)}$ denotes the hidden vector of the parent node of node k in DT-RNN. Hence the lower nodes receive error from both the CRF output and error propagation from parent node. The parameters within DT-RNN, Θ_{RNN}, are updated by applying chain rule with respect to updates of hidden vectors, and aggregating among all associated nodes, as shown in (8). The overall procedure of RNCRF is summarized in Algorithm 1.

5 Discussion

The best performing system (Toh and Wang, 2014) for SemEval challenge 2014 task 4 (subtask 1) employed CRFs with extensive hand-crafted features including those induced from dependency trees. However, their experiments showed that the addition of the features induced from dependency relations does not improve the performance. This indicates

Algorithm 1 Recursive Neural CRFs

```
Input: A set of customer review sequences: $S = \{s_1, ..., s_N\}$, and feature vectors of $d$ dimensions for each word $\{x_w\}$'s, window size $T$ for CRFs
Output: Parameters: $\Theta = \{\Theta_{\text{RNN}}, \Theta_U, V\}$
Initialization: Initialize $W_e$ using word2vec. Initialize $W_v$ and $\{W_r\}$'s randomly with uniform distribution between $\left[-\frac{\sqrt{6}}{\sqrt{2d+1}}, \frac{\sqrt{6}}{\sqrt{2d+1}}\right]$. Initialize $W_0$, $\{W_{+t}\}$'s, $\{W_{-t}\}$'s, $V$, and $b$ with all 0's
for each sentence $s_i$ do
    1: Use DT-RNN (1) to generate $\mathbf{h}_i$
    2: Compute $p(\mathbf{y}_i|\mathbf{h}_i)$ using (2)
    3: Use the backpropagation algorithm to update parameters $\Theta$ through (4)-(8)
end for
```

the difficulty of incorporating dependency structure explicitly as input features, which motivates the design of our model to use DT-RNN to encode dependency between words for feature learning. The most important advantage of RNCRF is the ability to learn the underlying dual propagation between aspect and opinion terms from the tree structure itself. Specifically as shown in Figure 1(c), where the aspect is *food* and the opinion expression is *like*. In the dependency tree, *food* depends on *like* with the relation *DOBJ*. During training, RNCRF computes the hidden vector h_{like} for *like*, which is also obtained from h_{food}. As a result, the prediction for *like* is affected by h_{food}. This is one-way propagation from *food* to *like*. During backpropagation, the error for *like* is propagated through a top-down manner to revise the representation h_{food}. This is the other-way propagation from *like* to *food*. Therefore, the dependency structure together with the learning approach help to enforce the dual propagation of aspect-opinion pairs as long as the dependency relation exists, either directly or indirectly.

5.1 Adding Linguistic/Lexicon Features

RNCRF is an end-to-end model, where feature engineering is not necessary. However, it is flexible to incorporate light hand-crafted features into RNCRF to further boost its performance, such as features from POS tags, name-list, or sentiment lexicon. These features could be appended to the hidden vector of each word, but keep fixed during training, unlike learnable neural inputs and the CRF weights as described in Section 4.3. As will be shown in experiments, RNCRF without any hand-crafted features slightly outperforms the best performing systems that involve heavy feature engineering efforts, and RNCRF with light feature engineering can achieve even better performance.

Domain	Training	Test	Total
Restaurant	3,041	800	3,841
Laptop	3,045	800	3,845
Total	6,086	1,600	7,686

Table 1: SemEval Challenge 2014 task 4 dataset

6 Experiment

6.1 Dataset and Experimental Setup

We evaluate our model on the dataset from SemEval Challenge 2014 task 4 (subtask 1), which includes reviews from two domains: restaurant and laptop[3]. The detailed description of the dataset is given in Table 1. As the original dataset only includes manually annotate labels for aspect terms but not for opinion terms, we manually annotated opinion terms for each sentence by ourselves to facilitate our experiments.

For word vector initialization, we train word embeddings with word2vec (Mikolov et al., 2013) on the Yelp Challenge dataset[4] for the restaurant domain and on the Amazon reviews dataset[5] (McAuley et al., 2015) for the laptop domain. The Yelp dataset contains 2.2M restaurant reviews with 54K vocabulary size. For the Amazon reviews, we only extracted the electronic domain that contains 1M reviews with 590K vocabulary size. We vary different dimensions for word embeddings and chose 300 for both domains. Empirical sensitivity studies on different dimensions of word embeddings are also conducted. Dependency trees are generated using Stanford Dependency Parser (Klein and Manning, 2003). Regarding CRFs, we implement a linear-chain CRF using CRFSuite (Okazaki, 2007). Because of the relatively small size of training data and a large number of parameters, we perform pre-training on the parameters of DT-RNN with cross-

[3] Experiments with more publicly available datasets, e.g. restaurant review dataset from SemEval Challenge 2015 task 12 will be conducted in our future work.

[4] http://www.yelp.com/dataset_challenge

[5] http://jmcauley.ucsd.edu/data/amazon/links.html

entropy error, which is a common strategy for deep learning (Erhan et al., 2009). We implement mini-batch stochastic gradient descent (SGD) with a batch size of 25, and an adaptive learning rate (AdaGrad) initialized at 0.02 for pretraining of DT-RNN, which runs 4 epochs for the restaurant domain and 5 epochs for the laptop domain. For parameter learning of the joint model RNCRF, we implement SGD with a decaying learning rate initialized at 0.02. We also try with varying context window size, and use 3 for the laptop domain and 5 for the restaurant domain, respectively. All parameters are chosen by cross validation.

As discussed in Section 5.1, hand-crafted features can be easily incorporated into RNCRF. We generate three types of simple features based on POS tags, name-list and sentiment lexicon to show further improvement by incorporating these features. Following (Toh and Wang, 2014), we extract two sets of name list from the training data for each domain, where one includes high-frequency aspect terms, and the other includes high-probability aspect words. These two sets are used to construct two lexicon features, i.e. we build a 2D binary vector: if a word is in a set, the corresponding value is 1, otherwise 0. For POS tags, we use Stanford POS tagger (Toutanova et al., 2003), and convert them to universal POS tags that have 15 different categories. We then generate 15 one-hot POS tag features. For sentiment lexicon, we use the collection of commonly used opinion words (around 6,800) (Hu and Liu, 2004a). Similar to name list, we create a binary feature to indicate whether the word belongs to opinion lexicon. We denote by RNCRF+F the proposed model with the three types of features.

Compared to the winning systems of SemEval Challenge 2014 task 4 (subtask 1), RNCRF or RNCRF+F uses additional labels of opinion terms for training. Therefore, to conduct fair comparison experiments with the winning systems, we implement RNCRF-O by omitting opinion labels to train our model (i.e., labels become “BA”, “IA”, “O”). Accordingly, we denote by RNCRF-O+F the RNCRF-O model with the three additional types of hand-crafted features.

6.2 Experimental Results

We compare our model with several baselines:

- **CRF-1**: a linear-chain CRF with standard linguistic features including word string, stylistics, POS tag, context string, and context POS tags.
- **CRF-2**: a linear-chain CRF with both standard linguistic features and dependency information including head word, dependency relations with parent token and child tokens.
- **LSTM**: an LSTM network built on top of word embeddings proposed by (Liu et al., 2015). We keep original settings in (Liu et al., 2015) but replace their word embeddings with ours (300 dimension). We try different hidden layer dimensions (50, 100, 150, 200) and reported the best result with size 50.
- **LSTM+F**: the above LSTM model with the three additional types of hand-crafted features as with RNCRF.
- **SemEval-1**, **SemEval-2**: the top two winning systems for SemEval challenge 2014 task 4 (subtask 1).
- **WDEmb+B+CRF**[6]: the model proposed by (Yin et al., 2016) using word and dependency path embeddings combined with linear context embedding features, dependency context embedding features and hand-crafted features (i.e., feature engineering) as CRF input.

The comparison results are shown in Table 2 for both the restaurant domain and the laptop domain. Note that we provide the same annotated dataset (both aspect labels and opinion labels are included for training) for CRF-1, CRF-2 and LSTM for fair comparison. It is clear that our proposed model RNCRF achieves superior performance compared with most of the baseline models. The performance is even better by adding simple hand-crafted features, i.e., RNCRF+F, with 0.92% and 3.87% absolute improvement over the best system in the challenge for aspect extraction for the restaurant domain and the laptop domain, respectively. This shows the advantage of

[6] We report the best results from the original paper (Yin et al., 2016).

	Restaurant		Laptop	
Models	Aspect	Opinion	Aspect	Opinion
SemEval-1	84.01	-	74.55	-
SemEval-2	83.98	-	73.78	-
WDEmb+B+CRF	**84.97**	-	75.16	-
CRF-1	77.00	78.95	66.21	71.78
CRF-2	78.37	78.65	68.35	70.05
LSTM	81.15	80.22	72.73	74.98
LSTM+F	82.99	82.90	73.23	77.67
RNCRF-O	82.73	-	74.52	-
RNCRF-O+F	84.25	-	77.26	-
RNCRF	84.05	80.93	76.83	76.76
RNCRF+F	84.93	**84.11**	**78.42**	**79.44**

Table 2: Comparison results in terms of F1 scores.

	Restaurant		Laptop	
Models	Aspect	Opinion	Aspect	Opinion
DT-RNN+SoftMax	72.45	69.76	66.11	64.66
CRF+word2vec	82.57	78.83	63.62	56.96
RNCRF	84.05	80.93	76.83	76.76
RNCRF+POS	84.08	81.48	77.04	77.45
RNCRF+NL	84.24	81.22	78.12	77.20
RNCRF+Lex	84.21	**84.14**	77.15	78.56
RNCRF+F	**84.93**	84.11	**78.42**	**79.44**

Table 3: Impact of different components.

combining high-level continuous features and discrete hand-crafted features. Though CRFs usually show promising results in sequence tagging problems, they fail to achieve comparable performance when lacking of extensive features (e.g., CRF-1). By adding dependency information explicitly in CRF-2, the result only improves slightly for aspect extraction. Alternatively, by incorporating dependency information into a deep learning model (e.g., RNCRF), the result shows more than 7% improvement for aspect extraction and 2% for opinion extraction.

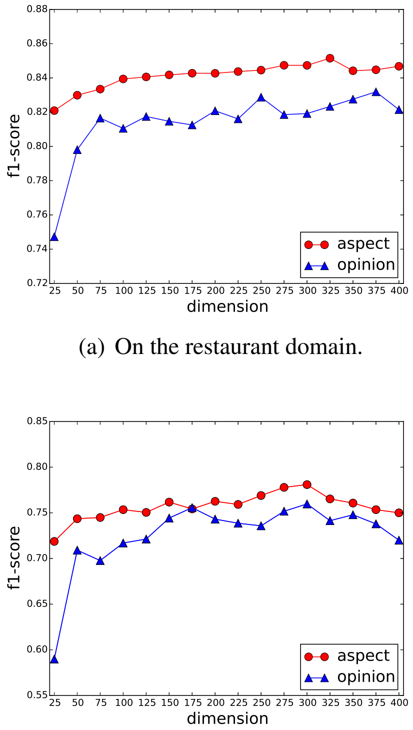
By removing the labels for opinion terms, RNCRF-O produces inferior results than RNCRF because the effect of dual propagation of aspect and opinion pairs disappears with the absence of opinion labels. This verifies our previous assumption that DT-RNN could learn the interactive effects within aspects and opinions. However, the performance of RNCRF-O is still comparable to the top systems and even better with the addition of simple linguistic features: 0.24% and 2.71% superior than the best system in the challenge for the restaurant domain and the laptop domain, respectively. This shows the robustness of our model even without additional opinion labels.

LSTM has shown comparable results for aspect extraction (Liu et al., 2015). However, in their work, they used well-pretrained word embeddings by training with large corpus or extensive external resources, e.g., chunking, and NER. To compare their model with RNCRF, we re-implement LSTM with the same word embedding strategy and labeling resources as ours. The results show that our model outperforms LSTM in aspect extraction by 2.90% and 4.10% for the restaurant domain and the laptop domain, respectively. We conclude that a standard LSTM model fails to extract the relations between aspect and opinion terms. Even with the addition of same linguistic features, LSTM is still inferior than RNCRF itself in terms of aspect extraction. Moreover, our result is comparable with WDEmb+B+CRF in the restaurant domain and better in the laptop domain (+3.26%). Note that WDEmb+B+CRF appended dependency context information into CRF while our model encode such information into high-level representation learning.

To test the impact of each component of RNCRF and the three types of hand-crafted features, we conduct experiments on different model settings:

- **DT-RNN+SoftMax**: rather than using a CRF, a softmax classifier is used on top of DT-RNN.
- **CRF+word2vec**: a linear-chain CRF with word embeddings only without using DT-RNN.
- **RNCRF+POS/NL/Lex**: the RNCRF model with POS tag or name list or sentiment lexicon feature(s).

The comparison results are shown in Table 3. Similarly, both aspect and opinion term labels are provided for training for each of the above models. Firstly, RNCRF achieves much better results compared to DT-RNN+SoftMax (+11.60% and +10.72% for the restaurant domain and the laptop domain in aspect extraction). This is because DT-RNN fails to fully exploit context information for sequence labeling, which, in contrast, can be achieved by CRF. Secondly, RNCRF outperforms CRF+word2vec, which proves the importance of



(a) On the restaurant domain.

(b) On the laptop domain.

Figure 3: Sensitivity studies on word embeddings.

DT-RNN for modeling interactions between aspects and opinions. Hence, the combination of DT-RNN and CRF inherits the advantages from both models. Moreover, by separately adding hand-crafted features, we can observe that name-list-based features and the sentiment lexicon feature are most effective for aspect extraction and opinion extraction, respectively. This may be explained by the fact that name-list-based features usually contain informative evident for aspect terms and sentiment lexicon provides explicit indication about opinions.

Besides the comparison experiments, we also conduct sensitivity test for our proposed model in terms of word vector dimensions. We tested a set of different dimensions ranging from 25 to 400, with 25 increment. The sensitivity plot is shown in Figure 3. The performance for aspect extraction is smooth with different vector lengths for both domains. For restaurant domain, the result is stable when dimension is larger than or equal to 100, with the highest at 325. For the laptop domain, the best result is at dimension 300, but with relatively small variations. For opinion extraction, the performance reaches a good level when the dimension is larger than or equal to 75 for the restaurant domain and 125 for the laptop domain. This proves the stability and robustness of our model.

7 Conclusion

We have presented a joint model, RNCRF, that achieves the state-of-the-art performance for explicit aspect and opinion term extraction on a benchmark dataset. With the help of DT-RNN, high-level features can be learned by encoding the underlying dual propagation of aspect-opinion pairs. RNCRF combines the advantages of DT-RNNs and CRFs, and thus outperforms the traditional rule-based methods in terms of flexibility, because aspect terms and opinion terms are not only restricted to certain observed relations and POS tags. Compared to feature engineering methods with CRFs, the proposed model saves much effort in composing features, and it is able to extract higher-level features obtained from non-linear transformations.

Acknowledgements

This research is partially funded by the Economic Development Board and the National Research Foundation of Singapore. Sinno J. Pan thanks the support from Fuji Xerox Corporation through joint research on *Multilingual Semantic Analysis* and the NTU Singapore Nanyang Assistant Professorship (NAP) grant M4081532.020.

References

Li Dong, Furu Wei, Chuanqi Tan, Duyu Tang, Ming Zhou, and Ke Xu. 2014. Adaptive recursive neural network for target-dependent twitter sentiment classification. In *ACL*, pages 49–54.

Dumitru Erhan, Pierre-Antoine Manzagol, Yoshua Bengio, Samy Bengio, and Pascal Vincent. 2009. The difficulty of training deep architectures and the effect of unsupervised pre-training. In *AISTATS*, pages 153–160.

Xavier Glorot, Antoine Bordes, and Yoshua Bengio. 2011. Domain adaptation for large-scale sentiment classification: A deep learning approach. In *ICML*, pages 97–110.

Christoph Goller and Andreas Küchler. 1996. Learning task-dependent distributed representations by backpropagation through structure. In *ICNN*, pages 347–352.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Minqing Hu and Bing Liu. 2004a. Mining and summarizing customer reviews. In *KDD*, pages 168–177.

Minqing Hu and Bing Liu. 2004b. Mining opinion features in customer reviews. In *AAAI*, pages 755–760.

Ozan İrsoy and Claire Cardie. 2014. Opinion mining with deep recurrent neural networks. In *EMNLP*, pages 720–728.

Mohit Iyyer, Jordan L. Boyd-Graber, Leonardo Max Batista Claudino, Richard Socher, and Hal Daumé III. 2014. A neural network for factoid question answering over paragraphs. In *EMNLP*, pages 633–644.

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

Wei Jin and Hung Hay Ho. 2009. A novel lexicalized hmm-based learning framework for web opinion mining. In *ICML*, pages 465–472.

Nal Kalchbrenner, Edward Grefenstette, and Phil Blunsom. 2014. A convolutional neural network for modelling sentences. In *ACL*, pages 655–665.

Yoon Kim. 2014. Convolutional neural networks for sentence classification. In *EMNLP*, pages 1746–1751.

Dan Klein and Christopher D. Manning. 2003. Accurate unlexicalized parsing. In *ACL*, pages 423–430.

John D. Lafferty, Andrew McCallum, and Fernando C. N. Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, pages 282–289.

Himabindu Lakkaraju, Richard Socher, and Christopher D. Manning. 2014. Aspect specific sentiment analysis using hierarchical deep learning. In *NIPS Workshop on Deep Learning and Representation Learning*.

Quoc V. Le and Tomas Mikolov. 2014. Distributed representations of sentences and documents. In *ICML*, pages 1188–1196.

Fangtao Li, Chao Han, Minlie Huang, Xiaoyan Zhu, Ying-Ju Xia, Shu Zhang, and Hao Yu. 2010. Structure-aware review mining and summarization. In *COLING*, pages 653–661.

Kang Liu, Liheng Xu, and Jun Zhao. 2012. Opinion target extraction using word-based translation model. In *EMNLP-CoNLL*, pages 1346–1356.

Kang Liu, Liheng Xu, Yang Liu, and Jun Zhao. 2013a. Opinion target extraction using partially-supervised word alignment model. In *IJCAI*, pages 2134–2140.

Qian Liu, Zhiqiang Gao, Bing Liu, and Yuanlin Zhang. 2013b. A logic programming approach to aspect extraction in opinion mining. In *WI*, pages 276–283.

Pengfei Liu, Shafiq Joty, and Helen Meng. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *EMNLP*, pages 1433–1443.

Bing Liu. 2011. *Web Data Mining: Exploring Hyperlinks, Contents, and Usage Data. Second Edition*. Data-Centric Systems and Applications. Springer.

Yue Lu, Malu Castellanos, Umeshwar Dayal, and ChengXiang Zhai. 2011. Automatic construction of a context-aware sentiment lexicon: An optimization approach. In *WWW*, pages 347–356.

Tengfei Ma and Xiaojun Wan. 2010. Opinion target extraction in Chinese news comments. In *COLING*, pages 782–790.

Julian McAuley, Jure Leskovec, and Dan Jurafsky. 2012. Learning attitudes and attributes from multi-aspect reviews. In *ICDM*, pages 1020–1025.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. 2015. Image-based recommendations on styles and substitutes. In *SIGIR*, pages 43–52.

Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *CoRR*, abs/1301.3781.

Nir Ofek, Soujanya Poria, Lior Rokach, Erik Cambria, Amir Hussain, and Asaf Shabtai. 2016. Unsupervised commonsense knowledge enrichment for domain-specific sentiment analysis. *Cognitive Computation*, 8(3):467–477.

Naoaki Okazaki. 2007. CRFsuite: a fast implementation of conditional random fields (CRFs). http://www.chokkan.org/software/crfsuite/.

Bo Pang and Lillian Lee. 2008. Opinion mining and sentiment analysis. *Foundations and Trends in Information Retrieval*, 2(1-2).

Maria Pontiki, Dimitris Galanis, John Pavlopoulos, Harris Papageorgiou, Ion Androutsopoulos, and Suresh Manandhar. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval*, pages 27–35.

Ana-Maria Popescu and Oren Etzioni. 2005. Extracting product features and opinions from reviews. In *EMNLP*, pages 339–346.

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.

Richard Socher, Christopher D. Manning, and Andrew Y. Ng. 2010. Learning Continuous Phrase Representations and Syntactic Parsing with Recursive Neural Networks. In *NIPS*, pages 1–9.

Richard Socher, Cliff C. Lin, Andrew Y. Ng, and Christopher D. Manning. 2011a. Parsing natural scenes and natural language with recursive neural networks. In *ICML*, pages 129–136.

Richard Socher, Jeffrey Pennington, Eric H. Huang, Andrew Y. Ng, and Christopher D. Manning. 2011b. Semi-Supervised Recursive Autoencoders for Predicting Sentiment Distributions. In *EMNLP*, pages 151–161.

Richard Socher, Brody Huval, Christopher D. Manning, and Andrew Y. Ng. 2012. Semantic Compositionality Through Recursive Matrix-Vector Spaces. In *EMNLP*, pages 1201–1211.

Richard Socher, Alex Perelygin, Jean Wu, Jason Chuang, Christopher D. Manning, Andrew Y. Ng, and Christopher Potts. 2013. Recursive deep models for semantic compositionality over a sentiment treebank. In *EMNLP*, pages 1631–1642.

Richard Socher, Andrej Karpathy, Quoc V. Le, Christopher D. Manning, and Andrew Y. Ng. 2014. Grounded compositional semantics for finding and describing images with sentences. *TACL*, 2:207–218.

Duyu Tang, Bing Qin, Xiaocheng Feng, and Ting Liu. 2015. Target-dependent sentiment classification with long short term memory. *CoRR*, abs/1512.01100.

Ivan Titov and Ryan T. McDonald. 2008. A joint model of text and aspect ratings for sentiment summarization. In *ACL*, pages 308–316.

Zhiqiang Toh and Wenting Wang. 2014. DLIREC: Aspect term extraction and term polarity classification system. In *SemEval*, pages 235–240.

Kristina Toutanova, Dan Klein, Christopher D. Manning, and Yoram Singer. 2003. Feature-rich part-of-speech tagging with a cyclic dependency network. In *NAACL*, pages 173–180.

Hao Wang and Martin Ester. 2014. A sentiment-aligned topic model for product aspect rating prediction. In *EMNLP*, pages 1192–1202.

Hongning Wang, Yue Lu, and ChengXiang Zhai. 2011. Latent aspect rating analysis without aspect keyword supervision. In *KDD*, pages 618–626.

David Weiss, Chris Alberti, Michael Collins, and Slav Petrov. 2015. Structured training for neural network transition-based parsing. In *ACL-IJCNLP*, pages 323–333.

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

Yichun Yin, Furu Wei, Li Dong, Kaimeng Xu, Ming Zhang, and Ming Zhou. 2016. Unsupervised word and dependency path embeddings for aspect term extraction. In *IJCAI*, pages 2979–2985.

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

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