Recursive Neural Structural Correspondence Network for Cross-domain Aspect and Opinion Co-Extraction

Wenya Wang[†‡] **and Sinno Jialin Pan**[†]
[†]Nanyang Technological University, Singapore
[‡]SAP Innovation Center Singapore
{wa0001ya, sinnopan}@ntu.edu.sg

Abstract

Fine-grained opinion analysis aims to extract aspect and opinion terms from each sentence for opinion summarization. Supervised learning methods have proven to be effective for this task. However, in many domains, the lack of labeled data hinders the learning of a precise extraction model. In this case, unsupervised domain adaptation methods are desired to transfer knowledge from the source domain to any unlabeled target domain. In this paper, we develop a novel recursive neural network that could reduce domain shift effectively in word level through syntactic relations. We treat these relations as invariant "pivot information" across domains to build structural correspondences and generate an auxiliary task to predict the relation between any two adjacent words in the dependency tree. In the end, we demonstrate state-of-the-art results on three benchmark datasets.

1 Introduction

The problem of fine-grained opinion analysis involves extraction of opinion targets (or aspect terms) and opinion expressions (or opinion terms) from each review sentence. For example, in the sentence: "*They offer good appetizers*", the aspect and opinion terms are *appetizers* and *good* correspondingly. Many supervised deep models have been proposed for this problem (Liu et al., 2015; Yin et al., 2016; Wang et al., 2017), and obtained promising results. However, these methods fail to adapt well across domains, because the aspect terms from two different domains are usually disjoint, e.g., *laptop* v.s. *restaurant*, leading to large domain shift in the feature vector space. Though unsupervised methods (Hu and Liu, 2004; Qiu et al., 2011) can deal with data with few labels, their performance is unsatisfactory compared with supervised ones.

There have been a number of domain adaptation methods for coarse-grained sentiment classification problems across domains, where an overall sentiment polarity of a sentence or document is being predicted. Nevertheless, very few approaches exist for cross-domain fine-grained opinion analysis due to the difficulties in fine-grained adaptation, which is more challenging than coarse-grained problems. Li et al. (2012) proposed a bootstrap method based on the TrAdaBoost algorithm (Dai et al., 2007) to iteratively expand opinion and aspect lexicons in the target domain by exploiting source-domain labeled data and cross-domain common relations between aspect terms and opinion terms. However, their model requires a seed opinion lexicon in the target domain and pre-mined syntactic patterns as a bridge. Ding et al. (2017) proposed to use rules to generate auxiliary supervision on top of a recurrent neural network to learn domain-invariant hidden representation for each word. The performance highly depends on the quality of the manually defined rules and the prior knowledge of a sentiment lexicon. In addition, the recurrent structure fails to capture the syntactic interactions among words intrinsically for opinion extraction. The requirement for rules makes the above methods non-flexible.

In this paper, we propose a novel cross-domain Recursive Neural Network (RNN)[1] for aspect and opinion terms co-extraction across domains. Our motivations are twofold: 1) The dependency relations capture the interactions among different words. These relations are especially important for identifying aspect terms and opinion terms (Qiu et al., 2011; Wang et al., 2016), which are also domain-invariant within the same language. Therefore, they can be used as "pivot" information to

[1]Here, we use RNN to denote recursive neural networks, rather than recurrent neural networks.

bridge the gap between different domains. 2) Inspired by the idea of *structural learning* (Ando and Zhang, 2005), the success of target task depends on the ability of finding good predictive structures learned from other related tasks, e.g., structural correspondence learning (SCL) (Blitzer et al., 2006) for coarse-grained cross-domain sentiment classification. Here, we aim to generate an auxiliary task on dependency relation classification. Different from previous approaches, our auxiliary task and the target extraction task are of heterogeneous label spaces. We aim to integrate this auxiliary task with distributed relation representation learning into a recursive neural network.

Specifically, we generate a dependency tree for each sentence from the dependency parser and construct a unified RNN that integrates an auxiliary task into the computation of each node. The auxiliary task is to classify the dependency relation for each direct edge in the dependency tree by learning a relation feature vector. To reduce label noise brought by inaccurate parsing trees, we further propose to incorporate an autoencoder into the auxiliary task to group the relations into different clusters. Finally, to model the sequential context interaction, we develop a joint architecture that combines RNN with a sequential labeling model for aspect and opinion terms extraction. Extensive experiments are conducted to demonstrate the advantage of our proposed model.

2 Related Work

Existing works for single-domain aspect/opinion terms extraction include unsupervised methods based on association rule mining (Hu and Liu, 2004), syntactic rule propagation (Qiu et al., 2011) or topic modeling (Titov and McDonald, 2008; Lu et al., 2009; Zhang et al., 2010), as well as supervised methods based on extensive feature engineering with graphical models (Jin and Ho, 2009; Li et al., 2010) or deep learning (Liu et al., 2015; Zhang et al., 2015; Wang et al., 2017; Yin et al., 2016). Among exiting deep models, improved results are obtained using dependency relations (Yin et al., 2016; Wang et al., 2016), which indicates the significance of syntactic word interactions for target term extraction. In cross-domain setting, there are very few works for aspect/opinion terms extraction including a pipelined approach (Li et al., 2012) and a recurrent neural network (Ding et al., 2017). Both of the methods require manual construction of common and pivot syntactic patterns or rules, which are indicative of aspect or opinion words.

There have been a number of domain adaptation approaches proposed for coarse-grained sentiment classification. Among existing methods, one active line focuses on projecting original feature spaces of two domains into the same low-dimensional space to reduce domain shift using pivot features as a bridge (Blitzer et al., 2007; Pan et al., 2010; Bollegala et al., 2015; Yu and Jiang, 2016). Another line learns domain-invariant features via autoencoders (Glorot et al., 2011; Chen et al., 2012; Zhou et al., 2016). Our work is more related to the first line by utilizing pivot information to transfer knowledge across domains, but we integrate the idea into a unified deep structure that can fully utilize syntactic structure for domain adaptation in fine-grained sentiment analysis.

3 Problem Definition & Motivation

Our task is to extract opinion and aspect terms within each review sentence. We denote a sentence by a sequence of tokens $\mathbf{x} = (w_1, w_2, ..., w_n)$. The output is a sequence of token-level labels $\mathbf{y} = (y_1, y_2, ..., y_n)$, with $y_i \in \{\text{BA}, \text{IA}, \text{BO}, \text{IO}, \text{N}\}$ that represents beginning of an aspect (BA), inside of an aspect (IA), beginning of an opinion (BO), inside of an opinion (IO) or none of the above (N). A subsequence of labels started with "BA" and followed by "IA" indicates a multi-word aspect term. In unsupervised domain adaptation, we are given a set of labeled review sentences from a source domain $\mathcal{D}_S = \{(\mathbf{x}_{S_i}, \mathbf{y}_{S_i})\}_{i=1}^{n_S}$, and a set of unlabeled sentences from a target domain $\mathcal{D}_T = \{\mathbf{x}_{T_j}\}_{j=1}^{n_T}$. Our goal is to predict token-level labels on $\mathcal{D}_T$.

Existing works for cross-domain aspect and/or opinion terms extraction require hand-coded rules and a sentiment lexicon in order to transfer knowledge across domains. For example in Figure 1, given a review sentence "*They offer good appetizers*" in the source domain and "*The laptop has a nice screen*" in the target domain. If *nice* has been extracted as a common sentiment word, and "OPINION-amod-ASPECT" has been identified as a common syntactic pattern from the source domain, *screen* could be deduced as an aspect term using the identified syntactic pattern (Li et al., 2012). Similarly, Ding et al. (2017) used a set of predefined rules based on syntactic relations and a sentiment lexicon to generate auxiliary labels to learn high-level feature representations through a

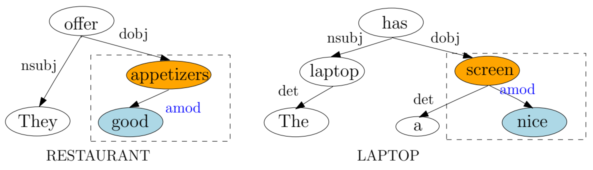


Figure 1: An example of two reviews with similar syntactic patterns.

recurrent neural network.

On one hand, these previous attempts have verified that syntactic information between words, which can be used as a bridge between domains, is crucial for domain adaptation. On the other hand, dependency-tree-based RNN (Socher et al., 2010) has proven to be effective to learn high-level feature representation of each word by encoding syntactic relations between aspect terms and opinion terms (Wang et al., 2016). With the above findings, we propose a novel RNN named **R**ecursive **N**eural **S**tructural **C**orrespondence **N**etwork (RNSCN) to learn high-level representation for each word across different domains. Our model is built upon dependency trees generated from a dependency parser. Different from previous approaches, we do not require any hand-coded rules or pre-selected pivot features to construct correspondences, but rather focus on the automatically generated dependency relations as the pivots. The model associates each direct edge in the tree with a relation feature vector, which is used to predict the corresponding dependency relation as an auxiliary task.

Note that the relation vector is the key in the model: it associates with the two interacting words and is used to construct structural correspondences between two different domains. Hence, the auxiliary task guides the learning of relation vectors, which in turn affects their correspondingly interactive words. Specifically in Figure 1, the relation vector for "amod" is computed from the features of its child and parent words, and also used to produce the hidden representation of its parent. For this relation path in both sentences, the auxiliary task enforces close proximity for these two relation vectors. This pushes the hidden representations for their parent nodes *appetizers* and *screen* closer to each other, provided that *good* and *nice* have similar representations. In a word, the auxiliary task bridges the gap between two different domains by drawing the words with similar syntactic properties closer to each other.

However, the relation vectors may be sensitive to the accuracy of the dependency parser. It might harm the learning process when some noise exists for certain relations, especially for informal texts. This problem of noisy labels has been addressed using perceptual consistency (Reed et al., 2015). Inspired by the taxonomy of dependency relations (de Marneffe and Manning, 2008), relations with similar functionalities could be grouped together, e.g., *dobj*, *iobj* and *pobj* all indicate objects. We propose to use an auto-encoder to automatically group these relations in an unsupervised manner. The reconstruction loss serves as the consistency objective that reduces label noise by aligning relation features with their intrinsic relation group.

4 Proposed Methodology

Our model consists of two components. The first component is a Recursive Neural Structural Correspondence Network (RNSCN), and the second component is a sequence labeling classifier. In this paper, we focus on Gated Recurrent Unit (GRU) as an implementation for the sequence labeling classifier. We choose GRU because it is able to deal with long-term dependencies compared to a simple Recurrent neural network and requires less parameters making it easier to train than LSTM. The resultant deep learning model is denoted by RNSCN-GRU. We also implement Conditional Random Field as the sequence labeling classifier, and denote the model by RNSCN-CRF accordingly.

The overall architecture of RNSCN-GRU without auto-encoder on relation denoising is shown in Figure 2. The left and right are two example sentences from the source and the target domain, respectively. In the first component, RNSCN, an auxiliary task to predict the dependency relation for each direct edge is integrated into a dependency-tree-based RNN. We generate a relation vector for each direct edge from its child node to parent node, and use it to predict the relation and produce the hidden representation for the parent node in the dependency tree. To address the issues of noisy relation labels, we further incorporate an auto-encoder into RNSCN, as will be shown in Figure 3.

While RNSCN mainly focuses on syntactic interactions among the words, the second component, GRU, aims to compute linear-context interactions. GRU takes the hidden representation of each word computed from RNSCN as inputs and further produces final representation of each word by taking linear contexts into consideration. We describe each component in detail in the following sections.

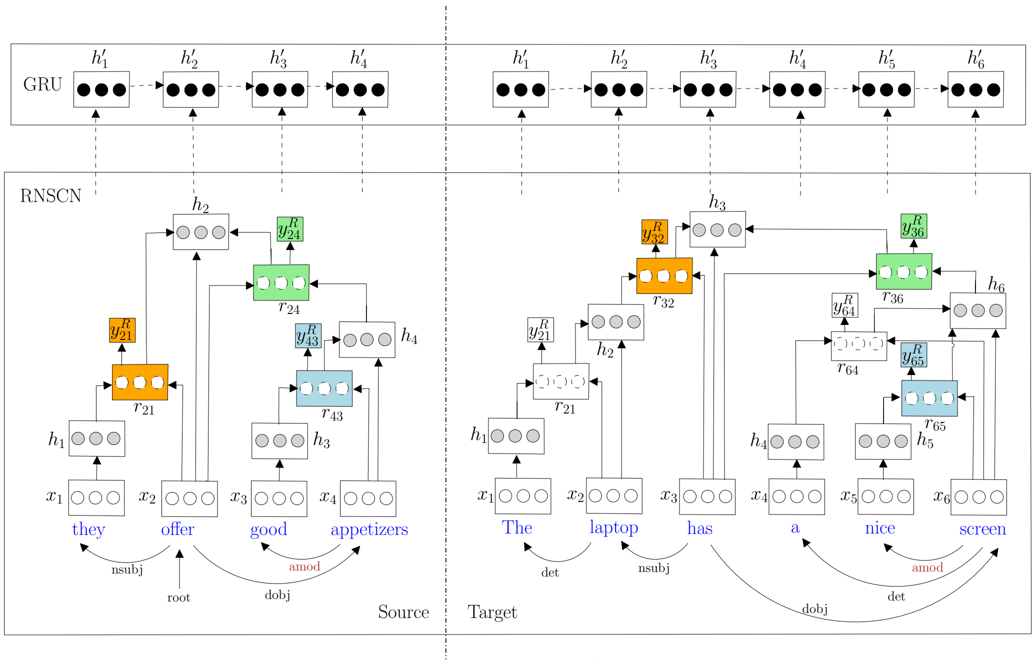


Figure 2: The architecture of RNSCN-GRU.

4.1 Recursive Neural Structural Correspondence Network

RNSCN is built on the dependency tree of each sentence, which is pre-generated from a dependency parser. Specifically, each node in the tree is associated with a word w_n, an input word embedding $\mathbf{x}_n \in \mathbb{R}^d$ and a transformed hidden representation $\mathbf{h}_n \in \mathbb{R}^d$. Each direct edge in the dependency tree associates with a relation feature vector $\mathbf{r}_{nm} \in \mathbb{R}^d$ and a true relation label vector $\mathbf{y}^R_{nm} \in \mathbb{R}^K$, where K is the total number of dependency relations, n and m denote the indices of the parent and child word of the dependency edge, respectively. Based on the dependency tree, the hidden representations are generated in a recursive manner from leaf nodes until reaching the root node. Consider the source-domain sentence shown in Figure 2 as an illustrative example, we first compute hidden representations for leaf nodes *they* and *good*:

$$\mathbf{h}_1 = \tanh(\mathbf{W}_x \mathbf{x}_1 + \mathbf{b}), \quad \mathbf{h}_3 = \tanh(\mathbf{W}_x \mathbf{x}_3 + \mathbf{b}),$$

where $\mathbf{W}_x \in \mathbb{R}^{d\times d}$ transforms word embeddings to hidden space. For non-leaf node *appetizer*, we first generate the relation vector $\mathbf{r}_{43}$ for the dependency edge $\mathbf{x}_4$ (appetizers) $\xrightarrow{\text{amod}}$ $\mathbf{x}_3$ (good) by

$$\mathbf{r}_{43} = \tanh(\mathbf{W}_h \mathbf{h}_3 + \mathbf{W}_x \mathbf{x}_4),$$

where $\mathbf{W}_h \in \mathbb{R}^{d\times d}$ transforms the hidden representation to the relation vector space. We then compute the hidden representation for *appetizer*:

$$\mathbf{h}_4 = \tanh(\mathbf{W}_{\text{amod}} \mathbf{r}_{43} + \mathbf{W}_x \mathbf{x}_4 + \mathbf{b}).$$

Moreover, the relation vector $\mathbf{r}_{43}$ is used for the auxiliary task on relation prediction:

$$\hat{\mathbf{y}}^R_{43} = \text{softmax}(\mathbf{W}_R \mathbf{r}_{43} + \mathbf{b}_R),$$

where $\mathbf{W}_R \in \mathbb{R}^{K\times d}$ is the relation classification matrix. The supervised relation classifier enforces close proximity of similar $\{\mathbf{r}_{nm}\}$'s in the distributed relation vector space. The relation features bridge the gap of word representations in different domains by incorporating them into the forward computations. In general, the hidden representation $\mathbf{h}_n$ for a non-leaf node is produced through

$$\mathbf{h}_n = \tanh\Big(\sum_{m\in\mathcal{M}_n} \mathbf{W}_{R_{nm}} \mathbf{r}_{nm} + \mathbf{W}_x \mathbf{x}_n + \mathbf{b}\Big), \quad (1)$$

where $\mathbf{r}_{nm} = \tanh(\mathbf{W}_h \cdot \mathbf{h}_m + \mathbf{W}_x \cdot \mathbf{x}_n)$, $\mathcal{M}_n$ is the set of child nodes of w_n, and $\mathbf{W}_{R_{nm}}$ is the relation transformation matrix tied with each relation R_{nm}. The predicted label vector $\hat{\mathbf{y}}^R_{nm}$ for $\mathbf{r}_{nm}$ is

$$\hat{\mathbf{y}}^R_{nm} = \text{softmax}(\mathbf{W}_R \cdot \mathbf{r}_{nm} + \mathbf{b}_R). \quad (2)$$

Here we adopt the the cross-entropy loss for relation classification between the predicted label vector $\hat{\mathbf{y}}^R_{nm}$ and the ground-truth $\mathbf{y}^R_{nm}$ to encode relation side information into feature learning:

$$\ell_R = \sum_{k=1}^{K} -\mathbf{y}^R_{nm[k]} \log \hat{\mathbf{y}}^R_{nm[k]}. \quad (3)$$

Through the auxiliary task, similar relations enforce participating words close to each other so that words with similar syntactic functionalities are clustered across domains. On the other hand, the pre-trained word embeddings group semantically-similar words. By taking them as input to RNN, together with the auxiliary task, our model encodes both semantic and syntactic information.

4.2 Reduce Label Noise with Auto-encoders

As discussed in Section 3, it might be hard to learn an accurate relation classifier when each class is a unique relation, because the dependency parser may generate incorrect relations as noisy labels. To address it, we propose to integrate an autoencoder into RNSCN. Suppose there is a set of latent groups of relations: $G = \{1, 2, ..., |G|\}$, where each relation belongs to only one group. For each relation vector, $\mathbf{r}_{nm}$, an autoencoder is performed before feeding it into the auxiliary classifier (2). The goal is to encode the relation vector to a probability distribution of assigning this relation to any group. As can be seen Figure 3, each relation vector $\mathbf{r}_{nm}$ is first passed through the autoencoder as follows,

$$p(G_{nm} = i|\mathbf{r}_{nm}) = \frac{\exp(\mathbf{r}_{nm}^\top \mathbf{W}_{enc}\mathbf{g}_i)}{\sum_{j\in G} \exp(\mathbf{r}_{nm}^\top \mathbf{W}_{enc}\mathbf{g}_j)}, \quad (4)$$

where G_{nm} denotes the inherent relation group for $\mathbf{r}_{nm}$, $\mathbf{g}_i \in \mathbb{R}^d$ represents the feature embedding for group i, and $\mathbf{W}_{enc} \in \mathbb{R}^{d\times d}$ is the encoding matrix that computes bilinear interactions between relation vector $\mathbf{r}_{nm}$ and relation group embedding $\mathbf{g}_i$. Thus, $p(G_{nm} = i|\mathbf{r}_{nm})$ represents the probability of $\mathbf{r}_{nm}$ being mapped to group i. An accumulated relation group embedding is computed as:

$$\mathbf{g}_{nm} = \sum_{i=1}^{|G|} p(G_{nm} = i|\mathbf{r}_{nm})\mathbf{g}_i. \quad (5)$$

For decoding, the decoder takes $\mathbf{g}_{nm}$ as input and tries to reconstruct the relation feature input $\mathbf{r}_{nm}$. Moreover, $\mathbf{g}_{nm}$ is also used as the higher-level feature vector for $\mathbf{r}_{nm}$ for predicting the relation label. Therefore, the objective for the auxiliary task in (3) becomes:

$$\ell_R = \ell_{R_1} + \alpha\ell_{R_2} + \beta\ell_{R_3}, \quad (6)$$

where

$$\ell_{R_1} = \|\mathbf{r}_{nm} - \mathbf{W}_{dec}\mathbf{g}_{nm}\|_2^2, \quad (7)$$

$$\ell_{R_2} = \sum_{k=1}^{K} -\mathbf{y}_{nm[k]}^R \log \hat{\mathbf{y}}_{nm[k]}^R, \quad (8)$$

$$\ell_{R_3} = \left\|\mathbf{I} - \bar{\mathbf{G}}^\top\bar{\mathbf{G}}\right\|_F^2. \quad (9)$$

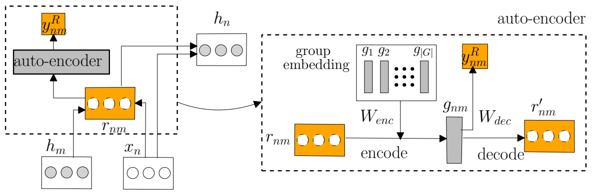


Figure 3: An autoencoder for relation grouping.

Here ℓ_{R_1} is the reconstruction loss with $\mathbf{W}_{dec}$ being the decoding matrix, ℓ_{R_2} follows (3) with $\hat{\mathbf{y}}_{nm}^R = \text{softmax}(\mathbf{W}_R\mathbf{g}_{nm} + \mathbf{b}_R)$ and ℓ_{R_3} is the regularization term on the correlations among latent groups with $\mathbf{I}$ being the identity matrix and $\bar{\mathbf{G}}$ being a normalized group embedding matrix that consists of normalized $\mathbf{g}_i$'s as column vectors. This regularization term enforces orthogonality between $\mathbf{g}_i$ and $\mathbf{g}_j$ for $i \neq j$. α and β are used to control the trade-off among different losses. With the auto-encoder, the auxiliary task of relation classification is conditioned on group assignment. The reconstruction loss further ensures the consistency between relation features and groupings, which is supposed to dominate classification loss when the observed labels are inaccurate. We denote RNSCN with auto-encoder by RNSCN$^+$.

4.3 Joint Models for Sequence Labeling

RNSCN or RNSCN$^+$ focuses on capturing and representing syntactic relations to build a bridge between domains and learn more powerful representations for tokens. However, it ignores the linear-chain correlations among tokens within a sentence, which is important for aspect and opinion terms extraction. Therefore, we propose a joint model, denoted by RNSCN-GRU (RNSCN$^+$-GRU), which integrates a GRU-based recurrent neural network on top of RNSCN (RNSCN$^+$), i.e., the input for GRU is the hidden representations $\mathbf{h}_n$ learned by RNSCN or RNSCN$^+$ for the n-th token in the sentence. For simplicity in presentation, we denote the computation of GRU by using the notation f_{GRU}. To be specific, by taking $\mathbf{h}_n$ as input, the final feature representation $\mathbf{h}'_n$ for each word is obtained through

$$\mathbf{h}'_n = f_{GRU}(\mathbf{h}'_{n-1}, \mathbf{h}_n; \mathbf{\Theta}), \quad (10)$$

where $\mathbf{\Theta}$ is the collection of the GRU parameters. The final token-level prediction is made through

$$\hat{\mathbf{y}}_n = \text{softmax}(\mathbf{W}_l \cdot \mathbf{h}'_n + \mathbf{b}_l), \quad (11)$$

where $\mathbf{W}_l \in \mathbb{R}^{5\times d'}$ transforms a d'-dimensional feature vector to class probabilities (note that we have 5 different classes as defined in Section 3).

The second joint model, namely RNSCN-CRF, combines a linear-chain CRF with RNSCN to learn the discriminative mapping from high-level features to labels. The advantage of CRF is to learn sequential interactions between each pair of adjacent words as well as labels and provide structural outputs. Formally, the joint model aims to output a sequence of labels with maximum conditional probability given its input. Denote by $\mathbf{y}$ a sequence of labels for a sentence and by $\mathbf{H}$ the embedding matrix for each sentence (each column denotes a hidden feature vector of a word in the sentence learned by RNSCN), the inference is computed as:

$$\begin{aligned}\hat{\mathbf{y}} &= \arg\max_{\mathbf{y}} p(\mathbf{y}|\mathbf{H}) \\ &= \arg\max_{\mathbf{y}} \frac{1}{Z(\mathbf{H})}\prod_{c\in C}\exp\langle \mathbf{W}_c, g(\mathbf{H}, \mathbf{y}_c)\rangle \quad (12)\end{aligned}$$

where C indicates the set of different cliques (unary and pairwise cliques in the context of linear-chain). $\mathbf{W}_c$ is tied for each different $\mathbf{y}_c$, which indicates the labels for clique c. The operator $\langle\cdot,\cdot\rangle$ is the element-wise multiplication, and $g(\cdot)$ produces the concatenation of $\{\mathbf{h}_n\}$'s in a context window of each word. The above two models both consider the sequential interaction of the words within each sentence, but the formalization and training are totally different. We will report the results for both joint models in the experiment section.

4.4 Training

Recall that in our cross-domain setting, the labels for terms extraction are only available in the source domain, but the auxiliary relation labels can be automatically produced for both domains via the dependency parser. Besides the source domain labeled data $\mathcal{D}_S = \{(\mathbf{x}_{S_i}, \mathbf{y}_{S_i})\}_{i=1}^{n_S}$, we denote by $\mathcal{D}_R = \{(\mathbf{r}_j, \mathbf{y}_j^R)\}_{j=1}^{n_R}$ the combined source and target domain data with auxiliary relation labels. For training, the total loss consists of token-prediction loss ℓ_S and relation-prediction loss ℓ_R:

$$\mathcal{L} = \sum_{\mathcal{D}_S} \ell_S(\mathbf{y}_{S_i}, \hat{\mathbf{y}}_{S_i}) + \gamma \sum_{\mathcal{D}_R} \ell_R(\mathbf{r}_j, \mathbf{y}_j^R), \quad (13)$$

where γ is the trade-off parameter, ℓ_S is the cross-entropy loss between the predicted extraction label in (11) and the ground-truth, and ℓ_R is defined in (6) for RNSCN$^+$ or (3) for RNSCN. For RNSCN-CRF, the loss becomes the negative log probability of the true label given the corresponding input:

$$\ell_S(\mathbf{y}_{S_i}, \hat{\mathbf{y}}_{S_i}) = -\log(\mathbf{y}_{S_i}|\mathbf{h}_{S_i}). \quad (14)$$

Dataset	Description	# Sentences	Training	Testing
R	Restaurant	5,841	4,381	1,460
L	Laptop	3,845	2,884	961
D	Device	3,836	2,877	959

Table 1: Data statistics with number of sentences.

The parameters for token-level predictions and relation-level predictions are updated jointly such that the information from the auxiliary task could be propagated to the target task to obtain better performance. This idea is in accordance with structural learning proposed by Ando and Zhang (2005), which shows that multiple related tasks are useful for finding the optimal hypothesis space. In our case, the set of multiple tasks includes the target terms extraction task and the auxiliary relation prediction task, which are closely related. The parameters are all shared across domains. The joint model is trained using back-propagation from the top layer of GRU or CRF to RNSCN until reaching to the input word embeddings in the bottom.

5 Experiments

5.1 Data & Experimental Setup

The data is taken from the benchmark customer reviews in three different domains, namely restaurant, laptop and digital devices. The restaurant domain contains a combination of restaurant reviews from SemEval 2014 task 4 subtask 1 (Pontiki et al., 2014) and SemEval 2015 task 12 subtask 1 (Pontiki et al., 2015). The laptop domain consists of laptop reviews from SemEval 2014 task 4 subtask 1. For digital device, we take reviews from (Hu and Liu, 2004) containing sentences from 5 digital devices. The statistics for each domain are shown in Table 1. In our experiments, we randomly split the data in each domain into training set and testing set with the proportion being 3:1. To obtain more rigorous result, we make three random splits for each domain and test the learned model on each split. The number of sentences for training and testing after each split is also shown in Table 1. Each sentence is labeled with aspect terms and opinion terms.

For each cross-domain task, we conduct both inductive and transductive experiments. Specifically, we train our model only on the training sets from both (labeled) source and (unlabeled) target domains. For testing, the inductive results are obtained using the test data from the target domain, and the transductive results are obtained using the (unlabeled) training data from the target domain.

The evaluation metric we used is F1 score. Following the setting from existing work, only exact match could be counted as correct.

For experimental setup, we use Stanford Dependency Parser (Klein and Manning, 2003) to generate dependency trees. There are in total 43 different dependency relations, i.e. 43 classes for the auxiliary task. We set the number of latent relation groups as 20. The input word features for RNSCN are pre-trained word embeddings using word2vec (Mikolov et al., 2013) which is trained on 3M reviews from the Yelp dataset[2] and electronics dataset in Amazon reviews[3] (McAuley et al., 2015). The dimension of word embeddings is 100. Because of the relatively small size of the training data compared with the number of parameters, we firstly pre-train RNSCN for 5 epochs with mini-batch size 30 and rmsprop initialized at 0.01. The joint model of RNSCN$^+$-GRU is then trained with rmsprop initialized at 0.001 and mini-batch size 30. The trade-off parameter α, β and γ are set to be 1, 0.001 and 0.1, respectively. The hidden-layer dimension for GRU is 50, and the context window size is 3 for input feature vectors of GRU. For the joint model of RNSCN-CRF, we implement SGD with a decaying learning rate initialized at 0.02. The context window size is also 3 in this case. Both joint models are trained for 10 epochs.

5.2 Comparison & Results

We compared our proposed model with several baselines and variants of the proposed model:

- **RNCRF**: A joint model of recursive neural network and CRF proposed by (Wang et al., 2016) for single-domain aspect and opinion terms extraction. We make all the parameters shared across domains for target prediction.
- **RNGRU**: A joint model of RNN and GRU. The hidden layer of RNN is taken as input for GRU. We share all the parameters across domains, similar to RNCRF.
- **CrossCRF**: A linear-chain CRF with hand-engineered features that are useful for cross-domain settings (Jakob and Gurevych, 2010), e.g., POS tags, dependency relations.
- **RAP**: The Relational Adaptive bootstraPping method proposed by (Li et al., 2012) that uses TrAdaBoost to expand lexicons.
- **Hier-Joint**: A recent deep model proposed by Ding et al. (2017) that achieves state-of-the-art performance on aspect terms extraction across domains.
- **RNSCN-GRU**: Our proposed joint model integrating auxiliary relation prediction task into RNN that is further combined with GRU.
- **RNSCN-CRF**: The second proposed model similar to RNSCN-GRU, which replace GRU with CRF.
- **RNSCN$^+$-GRU**: Our final joint model with auto-encoders to reduce auxiliary label noise.

Note that we do not implement other recent deep adaptation models for comparison (Chen et al., 2012; Yang and Hospedales, 2015), because Hier-Joint (Ding et al., 2017) has already demonstrated better performances than these models. The overall comparison results with the baselines are shown in Table 2 with average F1 scores and standard deviations over three random splits. Clearly, the results for aspect terms (AS) transfer are much lower than opinion terms (OP) transfer, which indicate that the aspect terms are usually quite different across domains, whereas the opinion terms could be more common and similar. Hence the ability to adapt the aspect extraction from the source domain to the target domain becomes more crucial. On this behalf, our proposed model shows clear advantage over other baselines for this more difficult transfer problem. Specifically, we achieve 6.77%, 5.88%, 10.55% improvement over the best-performing baselines for aspect extraction in R$\rightarrow$L, L$\rightarrow$D and D$\rightarrow$L, respectively. By comparing with RNCRF and RNGRU, we show that the structural correspondence network is indeed effective when integrated into RNN.

To show the effect of the integration of the autoencoder, we conduct experiments over different variants of the proposed model in Table 3. RNSCN-GRU represents the model without autoencoder, which achieves much better F1 scores on most experiments compared with the baselines in Table 2. RNSCN$^+$-GRU outperforms RNSCN-GRU in almost all experiments. This indicates the autoencoder automatically learns data-dependent groupings, which is able to reduce unnecessary label noise. To further verify that the autoencoder indeed reduces label noise when the parser is inaccurate, we generate new noisy parse trees by replacing some relations within each sentence with a random

[2] http://www.yelp.com/dataset_challenge
[3] http://jmcauley.ucsd.edu/data/amazon/links.html

Models	R→L		R→D		L→R		L→D		D→R		D→L	
	AS	OP	AS	OP	AS	OP	AS	OP	AS	OP	AS	OP
CrossCRF	19.72 (1.82)	59.20 (1.34)	21.07 (0.44)	52.05 (1.67)	28.19 (0.58)	65.52 (0.89)	29.96 (1.69)	56.17 (1.49)	6.59 (0.49)	39.38 (3.06)	24.22 (2.54)	46.67 (2.43)
RAP	25.92 (2.75)	62.72 (0.49)	22.63 (0.52)	54.44 (2.20)	46.90 (1.64)	67.98 (1.05)	34.54 (0.64)	54.25 (1.65)	45.44 (1.61)	60.67 (2.15)	28.22 (2.42)	59.79 (4.18)
Hier-Joint	33.66 (1.47)	- -	33.20 (0.52)	- -	48.10 (1.45)	- -	31.25 (0.49)	- -	47.97 (0.46)	- -	34.74 (2.27)	-
RNCRF	24.26 (3.97)	60.86 (3.35)	24.31 (2.57)	51.28 (1.78)	40.88 (2.09)	66.50 (1.48)	31.52 (1.40)	55.85 (1.09)	34.59 (1.34)	63.89 (1.59)	40.59 (0.80)	60.17 (1.20)
RNGRU	24.23 (2.41)	60.65 (1.04)	20.49 (2.68)	52.28 (2.69)	39.78 (0.61)	62.99 (0.95)	32.51 (1.12)	52.24 (2.37)	38.15 (2.82)	64.21 (1.11)	39.44 (2.79)	60.85 (1.25)
RNSCN-CRF	35.26 (1.31)	61.67 (1.35)	32.00 (1.48)	52.81 (1.29)	**53.38** (1.49)	67.60 (0.99)	34.63 (1.38)	56.22 (1.10)	48.13 (0.71)	65.06 (0.66)	46.71 (1.16)	61.88 (1.52)
RNSCN-GRU	37.77 (0.45)	62.35 (1.85)	33.02 (0.58)	57.54 (1.27)	53.18 (0.75)	71.44 (0.97)	35.65 (0.77)	60.02 (0.80)	**49.62** (0.34)	69.42 (2.27)	45.92 (1.14)	63.85 (1.97)
RNSCN$^+$-GRU	**40.43** (0.96)	**65.85** (1.50)	**35.10** (0.62)	**60.17** (0.75)	52.91 (1.82)	**72.51** (1.03)	**40.42** (0.70)	**61.15** (0.60)	48.36 (1.14)	**73.75** (1.76)	**51.14** (1.68)	**71.18** (1.58)

Table 2: Comparisons with different baselines.

Models	R→L		R→D		L→R		L→D		D→R		D→L	
	AS	OP	AS	OP	AS	OP	AS	OP	AS	OP	AS	OP
RNSCN-GRU	37.77	62.35	33.02	57.54	**53.18**	71.44	35.65	60.02	**49.62**	69.42	45.92	63.85
RNSCN-GRU (r)	32.97	50.18	26.21	53.58	35.88	65.73	32.87	57.57	40.03	67.34	40.06	59.18
RNSCN$^+$-GRU	**40.43**	**65.85**	**35.10**	**60.17**	52.91	**72.51**	**40.42**	**61.15**	48.36	**73.75**	**51.14**	**71.18**
RNSCN$^+$-GRU (r)	39.27	59.41	33.42	57.24	45.79	69.96	38.21	59.12	45.36	72.84	50.45	68.05

Table 3: Comparisons with different variants of the proposed model.

		R→L		R→D		L→R		L→D		D→R		D→L	
		AS	OP	AS	OP	AS	OP	AS	OP	AS	OP	AS	OP
OUT	Hier-Joint	33.66	-	33.20	-	48.10	-	31.25	-	47.97	-	34.74	-
	RNSCN$^+$-GRU*	39.06	-	34.07	-	47.98	-	38.51	-	47.49	-	48.49	-
	RNSCN$^+$	31.60	**65.89**	24.37	60.01	39.58	71.03	34.40	60.47	41.02	71.23	45.54	69.00
	RNSCN$^+$-GRU	**40.43**	65.85	**35.10**	**60.17**	**52.91**	**72.51**	**40.42**	**61.15**	**48.36**	**73.75**	**51.14**	**71.18**
IN	Hier-Joint	32.41	-	29.79	-	47.04	-	31.26	-	**47.41**	-	33.80	-
	RNSCN$^+$-GRU*	40.34	-	30.75	-	48.69	-	37.40	-	46.49	-	48.50	-
	RNSCN$^+$	30.76	63.65	22.48	59.24	39.54	70.25	35.32	60.00	37.75	70.64	43.72	68.27
	RNSCN$^+$-GRU	**41.27**	**65.44**	**33.58**	**60.28**	**52.48**	**72.10**	**39.73**	**60.18**	47.10	**72.19**	**50.23**	**70.21**

Table 4: Comparisons with different transfer setting.

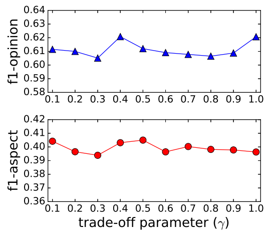
relation. Specifically, in each source domain, for each relation that connects to any aspect or opinion word, it has 0.5 probability of being replaced by any other relation. In Table 3, We denote the model with noisy relations with (r). Obviously, the performance of RNSCN-GRU without an autoencoder significantly deteriorates when the auxiliary labels are very noisy. On the contrary, RNSCN$^+$-GRU (r) achieves acceptable results compared to RNSCN$^+$-GRU. This proves that the autoencoder makes the model more robust to label noise and helps to adapt the information more accurately to the target data. Note that a large drop for $L \to R$ in aspect extraction might be caused by a large portion of noisy replacements for this particular data which makes it too hard to train a good classifier. This may not greatly influence opinion extraction, as shown, because the two domains usually share many common opinion terms. However, the significant difference in aspect terms makes the learning more dependent on common relations.

The above comparisons are made using the test data from target domains which are not available during training (i.e., the inductive setting). For more complete comparison, we also conduct experiments in the transductive setting. We pick our best model RNSCN$^+$-GRU, and show the effect of different components. To do that, we first remove the sequential structure on top, resulting in RNSCN$^+$. Moreover, we create another variant by removing opinion term labels to show the effect of the double propogation between aspect terms and opinion terms. The resulting model is named RNSCN$^+$-GRU*. As shown in Table 4, we denote by OUT and IN the inductive and transductive setting, respectively. The results shown are the average F1 scores among three splits[4]. In general, RNSCN$^+$-GRU shows similar performances for both inductive and transductive settings. This indicates the

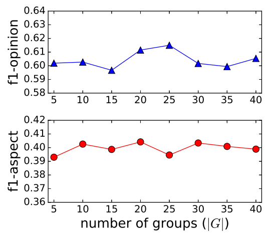
[4]We omit standard deviation here due to the limit of space.

G	Word
1	this, the, their, my, here, it, I, our, not
2	quality, jukebox, maitre-d, sauces, portions, volume, friend, noodles, calamari
3	in, slightly, often, overall, regularly, since, back, much, ago
4	handy, tastier, white, salty, right, vibrant, first, ok
5	get, went, impressed, had, try, said, recommended, call, love
6	is, are, feels, believes, seems, like, will, would

Table 5: Case studies on word clustering

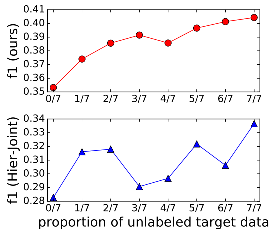


(a) On trade-off parameter.

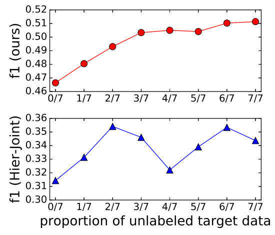


(b) On number of groups.

Figure 4: Sensitivity studies for L→D.



(a) F1-aspect on R→L



(b) F1-aspect on D→L

Figure 5: F1 vs proportion of unlabeled target data.

robustness and the ability to learn well when test data is not presented during training. Without opinion labels, RNSCN$^+$-GRU* still achieves better results than Hier-Joint most of the time. Its lower performance compared to RNSCN$^+$-GRU also indicates that in the cross-domain setting, the dual information between aspects and opinions is beneficial to find appropriate and discriminative relation feature space. Finally, the results for RNSCN$^+$ by removing GRU are lower than the joint model, which proves the importance of combining syntactic tree structure with sequential modeling.

To qualitatively show the effect of the auxiliary task with auto-encoders for clustering syntactically similar words across domains, we provide some case studies on the predicted groups of some words in Table 5. Specifically, for each relation in the dependency tree, we use (4) to obtain the most probable group to assign the word in the child node. The left column shows the predicted group index with the right column showing the corresponding words. Clearly, the words in the same group have similar syntactic functionalities, whereas the word types vary across groups.

In the end, we verify the robustness and capability of the model by conducting sensitivity studies and experiments with varying number of unlabeled target data for training, respectively. Figure 4 shows the sensitivity test for L→D, which indicates that changing of the trade-off parameter γ or the number of groups $|G|$ does not affect the model's performance greatly, i.e., less than 1% for aspect extraction and 2% for opinion extraction. This proves that our model is robust and stable against small variations. Figure 5 compares the results of RNSCN$^+$-GRU with Hier-Joint when increasing the proportion of unlabeled target training data from 0 to 1. Obviously, our model shows steady improvement with the increasing number of unlabeled target data. This pattern proves our model's capability of learning from target domain for adaptation.

6 Conclusion

We propose a novel dependency-tree-based RNN, namely RNSCN (or RNSCN$^+$), for domain adaptation. The model integrates an auxiliary task into representation learning of nodes in the dependency tree. The adaptation takes place in a common relation feature space, which builds the structural correspondences using syntactic relations among the words in each sentence. We further develop a joint model to combine RNSCN/RNSCN$^+$ with a sequential labeling model for terms extraction.

Acknowledgements

This work is supported by NTU Singapore Nanyang Assistant Professorship (NAP) grant M4081532.020, MOE AcRF Tier-1 grant 2016-T1-001-159, and Fuji Xerox Corporation through joint research on Multilingual Semantic Analysis.

References

Rie Kubota Ando and Tong Zhang. 2005. A framework for learning predictive structures from multiple tasks and unlabeled data. *JMLR* 6:1817–1853.

John Blitzer, Mark Dredze, and Fernando Pereira. 2007. Biographies, bollywood, boomboxes and blenders:

Domain adaptation for sentiment classification. In *ACL*. pages 187–205.

John Blitzer, Ryan McDonald, and Fernando Pereira. 2006. Domain adaptation with structural correspondence learning. In *EMNLP*. pages 120–128.

Danushka Bollegala, Takanori Maehara, and Ken ichi Kawarabayashi. 2015. Unsupervised cross-domain word representation learning. In *ACL*. pages 730–740.

Minmin Chen, Zhixiang Xu, Kilian Q. Weinberger, and Fei Sha. 2012. Marginalized denoising autoencoders for domain adaptation. In *ICML*. pages 1627–1634.

Wenyuan Dai, Qiang Yang, Gui-Rong Xue, and Yong Yu. 2007. Boosting for transfer learning. In *ICML*. pages 193–200.

Marie C. de Marneffe and Christopher D. Manning. 2008. The stanford typed dependencies representation. In *CrossParser*. pages 1–8.

Ying Ding, Jianfei Yu, and Jing Jiang. 2017. Recurrent neural networks with auxiliary labels for cross-domain opinion target extraction. In *AAAI*. pages 3436–3442.

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

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

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.

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

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.

Fangtao Li, Sinno Jialin Pan, Ou Jin, Qiang Yang, and Xiaoyan Zhu. 2012. Cross-domain co-extraction of sentiment and topic lexicons. In *ACL*. pages 410–419.

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

Yue Lu, ChengXiang Zhai, and Neel Sundaresan. 2009. Rated aspect summarization of short comments. In *WWW*. pages 131–140.

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.

Sinno Jialin Pan, Xiaochuan Ni, Jian-Tao Sun, Qiang Yang, and Zheng Chen. 2010. Cross-domain sentiment classification via spectral feature alignment. In *WWW*. pages 751–760.

Maria Pontiki, Dimitris Galanis, Haris Papageorgiou, Suresh Manandhar, and Ion Androutsopoulos. 2015. SemEval-2015 task 12: Aspect based sentiment analysis. In *SemEval 2015*. pages 486–495.

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.

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

Scott E. Reed, Honglak Lee, Dragomir Anguelov, Christian Szegedy, Dumitru Erhan, and Andrew Rabinovich. 2015. Training deep neural networks on noisy labels with bootstrapping. In *ICLR 2015*.

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

Ivan Titov and Ryan McDonald. 2008. Modeling online reviews with multi-grain topic models. In *WWW*. pages 111–120.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *EMNLP*. pages 616–626.

Wenya Wang, Sinno Jialin Pan, Daniel Dahlmeier, and Xiaokui Xiao. 2017. Coupled multi-layer tensor network for co-extraction of aspect and opinion terms. In *AAAI*. pages 3316–3322.

Yongxin Yang and Timothy M. Hospedales. 2015. A unified perspective on multi-domain and multi-task learning. In *ICLR*.

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.

Jianfei Yu and Jing Jiang. 2016. Learning sentence embeddings with auxiliary tasks for cross-domain sentiment classification. In *EMNLP*. pages 236–246.

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.

Meishan Zhang, Yue Zhang, and Duy Tin Vo. 2015. Neural networks for open domain targeted sentiment. In *EMNLP*.

Guangyou Zhou, Zhiwen Xie, Jimmy Xiangji Huang, and Tingting He. 2016. Bi-transferring deep neural networks for domain adaptation. In *ACL*. pages 322–332.