Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17)

Recurrent Neural Networks with Auxiliary Labels for Cross-Domain Opinion Target Extraction

Ying Ding, Jianfei Yu, Jing Jiang
School of Information Systems
Singapore Management University
{ying.ding.2011,jfyu.2014,jingjiang}@smu.edu.sg

Abstract

Opinion target extraction is a fundamental task in opinion mining. In recent years, neural network based supervised learning methods have achieved competitive performance on this task. However, as with any supervised learning method, neural network based methods for this task cannot work well when the training data comes from a different domain than the test data. On the other hand, some rule-based unsupervised methods have shown to be robust when applied to different domains. In this work, we use rule-based unsupervised methods to create auxiliary labels and use neural network models to learn a hidden representation that works well for different domains. When this hidden representation is used for opinion target extraction, we find that it can outperform a number of strong baselines with a large margin.

Introduction

Opinion target extraction is a fundamental problem in opinion mining. Its goal is to extract from opinionated texts the targets on which opinions have been expressed. For example, given the sentence "I like the tuna sandwich and chicken salad very much," the opinion targets should be *tuna sandwich* and *chicken salad*. Opinion target extraction has many downstream applications such as sentiment classification and opinion summarization. Given its importance, the problem has attracted much attention in the research community in the last decade (Schouten and Frasincar 2016).

Opinion target extraction is typically modeled as a supervised sequence labeling problem. A traditional approach is to use hand-crafted discrete features at token level coupled with Conditional Random Fields (CRFs) (Lafferty, McCallum, and Pereira 2001) to extract opinion targets. In recent years, with the advances of deep learning techniques for NLP, many researchers also tried to apply deep neural networks to this problem. In particular, Recurrent Neural Networks (RNNs) and their variants such as BiRNNs and LSTMs have been applied to opinion target extraction and shown to be effective (Liu, Joty, and Meng 2015).

As with any supervised learning method, the neural network based methods for opinion target extraction also suffers from the domain adaptation problem. This happens when the training data labeled with opinion targets comes from a domain different from the test data. For example, the labeled training data may contain laptop reviews, but the test data may contain restaurant reviews. This domain difference often causes a serious problem because the opinion target words can be very different in the two domains. For example, in the restaurant domain, words such as *food* and *drink* are frequent opinion targets, but if the training data comes from the laptop domain, you may not see these words at all in the training data. In our experiments, we find that some state-of-the-art RNN based method for opinion target extraction may face a performance drop of up to 40% when the training and the test data comes from different domains.

On the other hand, people have also developed some rule-based methods for opinion target extraction that are unsupervised and therefore more robust in different domains. For example, the double propagation method developed by Qiu et al. (2011) is an unsupervised method that has been shown to work well without any training data.

In this paper, we study how we can take advantage of neural network based supervised methods and rule-based unsupervised methods to develop a method for cross-domain opinion target extraction. Our method is motivated by the idea of learning a good hidden representation for both the source and the target domains by utilizing syntactic rules that are domain-independent. Specifically, we first use these syntactic rules to generate *auxiliary* labels. We then use recurrent neural networks, in particular, long short-term memory networks (LSTMs), to learn a good hidden layer that works well for predicting these auxiliary labels for both the source and the target domains. In addition, this hidden layer is also trained to work well for predicting the opinion targets in the source domain. We propose two different neural network architectures to learn the hidden representations. Our experiments using reviews from four different domains show that our proposed method can significantly outperform a number of strong baselines, including some domain adaptation methods.

The main contribution of our work is a novel way of combining rule-based, unsupervised opinion target extraction with neural network based supervised models to achieve better performance in a cross-domain setting.

Related Work

Opinion Target Extraction (OTE): Most existing studies of opinion target extraction can be categorized into rule-based methods or supervised machine learning based methods. The former either focuses on highly frequent single and compound nouns (Hu and Liu 2004) or defines some syntactic rules to detect opinion targets (Qiu et al. 2011). The later usually treats the problem as a sequence labeling task by applying Hidden Markov Models (HMMs) (Jin and Ho 2009) or Conditional Random Fields (CRFs) (Yang and Cardie 2013). With the recent advances of deep learning techniques for NLP, many neural network based models such as Neural-CRF, RNNs and their variants have been applied to this problem and shown competitive performance (Liu, Joty, and Meng 2015; Zhang, Zhang, and Vo 2015; Yin et al. 2016; Wang et al. 2016). While these studies focus on standard single-domain opinion target extraction, our work aims at developing a general neural network based method for cross-domain opinion target extraction.

Cross-Domain OTE: To the best of our knowledge, only a few studies have explored cross-domain opinion target extraction by using hand-crafted domain-independent features and CRF models (Jakob and Gurevych 2010; Chernyshevich 2014). Different from them, our proposed models are based on distributed word vectors and neural networks.

Domain Adaptation: Domain adaptation has attracted much attention in recent years (Pan and Yang 2010). One typical line of work aims to derive a general low-dimensional cross-domain representation by leveraging either auxiliary tasks (Blitzer, McDonald, and Pereira 2006) or unsupervised auto-encoders (Chen et al. 2012). Another line of work focuses on inducing robust cross-domain feature embeddings based on predicting its neighboring features (Yang and Eisenstein 2015). Our work is similar to the first line of work since we try to learn a general cross-domain representation through auxiliary labels.

Methodology

Notation

Opinion target extraction aims at extracting all the opinion targets from a given sentence. An opinion target is not restricted to a single token; in fact, it often contains multiple tokens. The task is therefore a typical sequence labeling problem.

Formally, we represent a sentence as a sequence of tokens $\mathbf{x} = (w_1, w_2, \ldots, w_N)$, where each w_i is a word type from a vocabulary $\mathcal{V}$. Opinion targets are indicated by token-level labels $\mathbf{y} = (y_1, y_2, \ldots, y_N)$, where each $y_i \in \{B, I, O\}$. The three labels B, I and O refer to the beginning, inside and outside of an opinion target, respectively, and they follow the standard BIO notation used in sequence labeling. A sample review sentence together with its opinion target labels are shown in Figure 1.

We assume that there is a set of labeled review sentences from a *source* domain, denoted with $\mathcal{D}^s = \{(\mathbf{x}^s, \mathbf{y}^s)\}$. On the other hand, the sentences from which opinion targets need to be extracted come from a different *target* domain and are denoted with $\mathcal{D}^t = \{\mathbf{x}^t\}$. We would like to use both

sentence:	I	like	the	tuna	sandwich	and	chicken	salad	very	much	.
label:	O	O	O	B	I	O	B	I	O	O	O

Figure 1: A sample sentence and its labels.

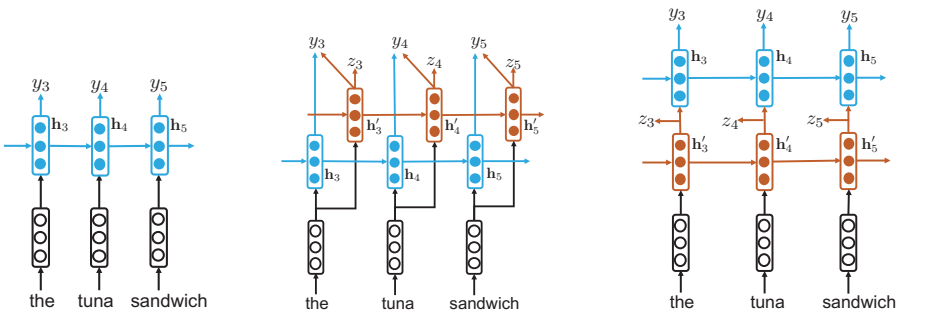
$\mathcal{D}^s$ and $\mathcal{D}^t$ to train a good model for opinion target extractions for the target domain.

Overview of Our Method

Our method is essentially a supervised method based on recurrent neural networks. The key to our method is a hidden layer of the neural network that is trained using auxiliary labels created by domain-independent rules. The idea of using auxiliary labels to induce a representation for domain adaptation is not new (Blitzer, McDonald, and Pereira 2006; Blitzer, Dredze, and Pereira 2007). It essentially follows the principle of multi-task learning, where it is generally believed that if multiple prediction tasks are related, then the underlying prediction models are likely to share some common feature structures. When the auxiliary tasks are related to the actual prediction task and the labels of the auxiliary tasks can be easily obtained for both the source and the target domains, we can use the auxiliary tasks to help us induce a good hidden feature representation that is good for domain adaptation.

Neural network models are intrinsically suitable for this kind of multi-task learning based domain adaptation because we can naturally use one of the hidden layers as the cross domain hidden representation. To the best of our knowledge, however, there has not been any work on extending neural network models to solve the domain adaptation problem for opinion target extraction. In our method, we use a Recurrent Neural Network (specifically, an LSTM) to process an input sentence such that each token has a corresponding hidden vector. Typically this hidden vector will then go through a linear transformation followed by a softmax layer to make the final prediction. In our method, this hidden vector is used for predicting not only the opinion target label but also some auxiliary label. The auxiliary labels are predicted by manually-crafted syntactic rules, which will be detailed below.

Figure 2 illustrates the main idea of our method at a high level. We can see that typically, as shown in Figure 2(a), the hidden layer $\mathbf{h}$ is learned only through back propagation from the true labels $\mathbf{y}$, which are only available in the source domain. With our method, as shown in Figure 2(b) and Figure 2(c), there is an auxiliary hidden layer $\mathbf{h}'$ that is learned using some auxiliary labels $\mathbf{z}$. And this auxiliary hidden layer $\mathbf{h}'$ is then either to be concatenated with $\mathbf{h}$ or to generate $\mathbf{h}$ in order to predict the true labels $\mathbf{y}$. Because the auxiliary labels $\mathbf{z}$ are available in both the source and the target domains, we can expect the auxiliary hidden layer $\mathbf{h}'$ to be properly learned such that it works well for both domains.



(a) Standard (b) Ours (Concatenated Network) (c) Ours (Hierarchical Network)

Figure 2: Overview of the standard method and our method.

Recurrent Neural Networks for Opinion Target Extraction

In this section, we describe how we use recurrent neural networks for opinion target extraction. Note that this is a standard approach and has been studied before (Liu, Joty, and Meng 2015). Recall that we use $\mathbf{x} = (w_1, w_2, \ldots, w_N)$ to represent an input sentence. Let $\mathbf{x}_i \in \mathbb{R}^d$ denote the embedding vector for word w_i. A recurrent neural network model aims at learning a hidden vector representation for each position i that supposedly encodes all the tokens from the beginning of the sentence up to position i. Specifically, let $\mathbf{h}_{i-1}$ denote such a hidden vector corresponding to position $i-1$. Then the hidden vector $\mathbf{h}_i$ is defined as follows:

$$\mathbf{h}_i \quad = \quad g(\mathbf{h}_{i-1}, \mathbf{x}_i),$$

where $g(\cdot)$ is some function. For example, in standard RNN, we have

$$\mathbf{h}_i \quad = \quad f(\mathbf{U}\mathbf{h}_{i-1} + \mathbf{V}\mathbf{x}_i + \mathbf{c}),$$

where $\mathbf{U} \in \mathbb{R}^{l \times l}$ and $\mathbf{V} \in \mathbb{R}^{l \times d}$ are weight matrices, $\mathbf{c} \in \mathbb{R}^l$ is a bias vector, l is the dimension of the hidden layer, and $f(\cdot)$ is an element-wise non-linear transformation function.

In our experiments, we experiment with a few different types of RNNs, including the standard RNN, bi-directional RNN, long short-term memory (LSTM) network (which is a special form of RNN), and bi-directional LSTM. We will not give the details of LSTM here. Interested readers can refer to Hochreiter and Schmidhuber (1997) for details.

To simplify the discussion, we use Θ to denote all the parameters used in any type of an RNN, and represent the hidden layer as

$$\mathbf{h}_i = \mathit{RNN}_\Theta(\mathbf{h}_{i-1}, \mathbf{x}_i).$$

The vector $\mathbf{h}_i$ is then used to predict the opinion target label as follows:

$$p(y_i \mid \mathbf{h}_i) \quad = \quad \text{softmax}(\mathbf{W}\mathbf{h}_i + \mathbf{b}),$$

where $\mathbf{W}$ is a weight matrix and $\mathbf{b}$ is a bias vector, both to be learned.

Rule-based Auxiliary Labels

Our preliminary experiments using the RNN model presented above on target opinion extraction suggest that the supervised RNN model relies much on lexical information. This is not surprising because the RNN model does not model syntactic structures of a sentence such as part-of-speech tags and dependency relations. Although lexical information is very important for opinion target extraction, it is also very domain specific. As a result, we find that the RNN model performs poorly in cross-domain settings.

On the other hand, people have studied how to use general syntactic patterns to detect opinion targets (Zhuang, Jing, and Zhu 2006; Qiu et al. 2011). An important observation is that opinion targets often co-occur with explicit opinion expressions, which usually contain opinion words. Syntactically, there are some patterns between opinion words and opinion targets, and these patterns tend to be general across different domains. For example, usually the object of the verb *love* is an opinion target. Using this rule, we can predict that the phrases *tuna sandwich* and *chicken salad* in the sentence "I love tuna sandwich and chicken salad very much" are opinion targets. We can also predict that the phrase *the design of iPhone 7* in the sentence "I love the design of iPhone 7" is an opinion target. We can see that the two sentences come from very different domains, but the rule is general.

Based on the work by Qiu et al. (2011), we develop a set of rules that use syntactic patterns to detect potential opinion targets. The rules are based on three dependency relations: *amod*, *nsubj* and *dobj*. In the descriptions below, we use arrows to indicate the direction of the dependency relations. We use `T` to denote a potential opinion target and `O` to denote an opinion word. We use four rules (R1, R2, R3 and R4) shown in Table 1 to identify `T`.

In addition, we have the constraints that the opinion word `O` must come from a pre-defined sentiment lexicon, and the POS tag of the target `T` must be one of {NN, NNS, NNP, NNPS}.

The rules above can only help us identify the head word of an opinion target. However, many opinion targets consist of multiple tokens. In order to identify additional tokens in opinion targets, we analyze the dependency relations within

RuleID	Rule	Example
R1	O $\xrightarrow{amod}$ T	They have nice dessert. (nice $\xrightarrow{amod}$ *dessert*)
R2	T $\xrightarrow{nsubj}$ O	Its camera is great. (*camera* $\xrightarrow{nsubj}$ great)
R3	T $\xrightarrow{dobj}$ O	I love their fries. (*fries* $\xrightarrow{dobj}$ love)
R4	T $\xrightarrow{nsubj}$ H $\xleftarrow{amod}$ O	iPhone is the best cellphone. (*iPhone* $\xrightarrow{nsubj}$ phone $\xleftarrow{amod}$ best)
ER1	W $\xrightarrow{amod}$ T	I like Indian food. (*Indian* $\xrightarrow{amod}$ food)
ER2	W $\xrightarrow{nn}$ T	Their spring roll is great. (*spring* $\xrightarrow{nn}$ roll)
ER3	$W_2 \xrightarrow{pobj} W_1 \xrightarrow{prep}$ T	I like the design of iPhone. (*iPhone* $\xrightarrow{pobj}$ *of* $\xrightarrow{prep}$ design)

Table 1: Rules for detecting opinion targets. H represents any word. W represents an additional target word to be detected using the expansion rules.

opinion targets and identify a set of expansion rules to expand opinion targets. They are shown as ER1, ER2 and ER3 in Table 1.

The rules described above can help us identify potential opinion targets in any given domain. However, using only these rules to extract opinion targets does not give very competitive results. This is because the coverage of these rules is still limited, and therefore the performance of a purely unsupervised method using these rules is not competitive, as we will see in the experiment section. In the next section, we will describe in detail how we combine these rules with recurrent neural network models to perform cross-domain opinion target extraction.

Two Architectures for Cross-domain Opinion Target Extraction

In this section, we will describe the neural network architectures we use to combine auxiliary labels with true labels in order to perform cross-domain opinion target extraction. The core of our models is to learn a hidden vector representation for each token that is useful for both the source and the target domains.

First of all, we have the following training data available to us. Recall that in the source domain, we have a set of sentences together with the true opinion target labels, denoted as $\mathcal{D}^s = \{(\mathbf{x}^s, \mathbf{y}^s)\}$. Next, for both sentences in the source domain and sentences in the target domain, based on the syntactic rules we have defined, we can obtain their auxiliary label sequences. Let us use $\mathbf{z}$ to denote the auxiliary labels of sentence $\mathbf{x}$. Let $\mathcal{D}^a = \{(\mathbf{x}^a, \mathbf{z}^a)\}$ denote all the sentences from the source and the target domains together with their auxiliary labels.

We now present two neural network architectures to use $\mathcal{D}^s$ and $\mathcal{D}^a$ to learn a prediction model. In both architectures, we introduce an auxiliary hidden layer $\mathbf{h}'$.

Concatenated Network In the first architecture, we first use an RNN to create the auxiliary hidden layer $\mathbf{h}'$ as follows:

$$\mathbf{h}'_i = RNN_{\Theta'}(\mathbf{h}'_{i-1}, \mathbf{x}_i). \quad (1)$$

This hidden layer will be used to predict the auxiliary labels:

$$p(z_i \mid \mathbf{h}'_i) = \text{softmax}(\mathbf{W}'\mathbf{h}'_i + \mathbf{b}'). \quad (2)$$

We also use a different RNN to create the standard hidden layer $\mathbf{h}$ as follows:

$$\mathbf{h}_i = RNN_{\Theta}(\mathbf{h}_{i-1}, \mathbf{x}_i).$$

Next, we concatenate $\mathbf{h}$ and $\mathbf{h}'$ into a single vector:

$$\overline{\mathbf{h}}_i = \mathbf{h}_i \bigoplus \mathbf{h}'_i.$$

This concatenated hidden vector is then used to predict the true opinion target label:

$$p(y_i \mid \overline{\mathbf{h}}_i) = \text{softmax}(\mathbf{W}\overline{\mathbf{h}}_i + \mathbf{b}).$$

This architecture is shown in Figure 2(b). We can see that different from a standard model, this model uses the additional auxiliary hidden vector $\mathbf{h}'$ together with $\mathbf{h}$ to predict the final labels.

Hierarchical Network In the second architecture, the auxiliary hidden vector $\mathbf{h}'$ is defined in the same way as in Eqn. (1), and the probability distribution $p(z_i \mid \mathbf{h}'_i)$ is also defined in the same way as in Eqn. (2). However, the standard hidden layer $\mathbf{h}$ now uses $\mathbf{h}'$ as input:

$$\mathbf{h}_i = RNN_{\Theta}(\mathbf{h}_{i-1}, \mathbf{h}'_i). \quad (3)$$

And finally, to predict the true opinion target label, we have

$$p(y_i \mid \mathbf{h}_i) = \text{softmax}(\mathbf{W}\mathbf{h}_i + \mathbf{b}). \quad (4)$$

This architecture is shown in Figure 2(c). We call this the hierarchical network because $\mathbf{h}'$ and $\mathbf{h}$ now reside at different layers of the neural network.

While both architectures can combine the power of domain-independent rules and the true opinion target labels from the source domain, they differ in how knowledge from these two parts are integrated. In the Concatenated Network, there is not much interaction between $\mathbf{h}'$ and $\mathbf{y}$. The Hierarchical Network has a more complicated mechanism by feeding $\mathbf{h}'$ into the RNN that produces $\mathbf{h}$. However, both models share the similar idea of (1) using auxiliary labels to encode domain-independent rules, and (2) learning parameters based on true annotated labels and auxiliary labels to obtain representation vectors that are potentially useful across different domains.

Learning the Parameters

To learn the parameters, we use the commonly-used log likelihood objective function. Note that there are two parts in our loss function, one related to the auxiliary labels $\mathbf{z}$ and the other related to the true labels $\mathbf{y}$.

Let us define the following loss functions:

$$L_z = \sum_{(\mathbf{x}^a, \mathbf{z}^a) \in \mathcal{D}^a} -\log p(\mathbf{z}^a \mid \mathbf{x}^a),$$

$$L_y = \sum_{(\mathbf{x}^s, \mathbf{y}^s) \in \mathcal{D}^s} -\log p(\mathbf{y}^s \mid \mathbf{x}^s).$$

To learn the parameters of our model, we can divide our training process into two steps by first minimizing L_z and then minimizing L_y with all the parameters learned by minimizing L_z fixed. We can also jointly minimize the sum of L_z and L_y with respect to all the parameters. We refer to the former as *separate* training and the latter as *joint* training. Back propagation is used in both training strategies. In our experiments, we will compare their performance.

Experiments

Experiment Settings

Datasets We use reviews from four different domains for our experiments. The four domains are restaurant, laptop, digital device and web service. The restaurant data is a combination of the restaurant reviews from SemEval 2014 (Pontiki et al. 2014) and SemEval 2015 (Pontiki et al. 2015). The laptop data comes from SemEval 2015 (Pontiki et al. 2015). The digital device dataset contains review sentences on five digital devices and was created by Kessler et al. (2010). The web service dataset was introduced by Toprak, Jakob, and Gurevych (2010) and consists of sentences from reviews of web services. The sentiment lexicon we use was downloaded from University of Illinois at Chicago.[1]

During preprocessing, all words are converted to lowercase, URL links are replaced with <URL> and numbers are replaced with <NUM>. We also remove some noisy sentences in the web service dataset. After preprocessing, the basic statistics of our datasets are shown in Table 2. For simplicity, we use Restaurant, Laptop, Device and Service to denote each of the datasets, respectively.

Dataset	# Sentences	# Words
Restaurant	5,841	88,707
Laptop	3,845	63,011
Device	3,836	70,913
Service	8,545	159,742

Table 2: Basic statistics of the datasets.

Evaluation Metric We use the $F1$ score of opinion targets as the evaluation metric. Following previous work (Liu, Joty, and Meng 2015; Zhang, Zhang, and Vo 2015), we only consider exact matches, which means a target is considered correctly extracted only if the output of a model is exactly the same without any missing word or extra word.

Parameter Settings We use pre-trained word embeddings from Google word2vec[2] to initialize the word embeddings in our models. The word embeddings are updated in the training process. To learn the parameters of our model, we use the Adagrad algorithm with a mini-batch size of 10 sentences. The initial learning rate is set to 0.01. Following previous work (Liu, Joty, and Meng 2015; Zhang, Zhang, and Vo 2015; Wang et al. 2016), we concatenate the word embeddings of the current word, its previous word and next word as the input to our model. This setting is also used in our neural network baselines. For each target domain, we leave out 200 randomly selected sentences as validation set. The dimension of the hidden layers is determined according to the performance on the validation set. We find 100 to be the best and the reported results below are obtained using 100 as the hidden layer dimension. All neural network models are trained for 15 iterations. Based on performance on the validation set, we choose the best model across the 15 iterations as our final model.

Models for Comparison

We use the following baselines for comparison:

- **CRF**: This is a traditional sequence labeling model using Conditional Random Field and discrete features such as word types, POS tags and dependency relations. It has been used for both single-domain and cross-domain opinion target extraction (Jakob and Gurevych 2010).
- **mDA**: This is a recently proposed domain adaptation method using marginalized denoising autoencoders (Chen et al. 2012). We use features that have been proven to be useful for opinion target extraction, including current word, previous word, next word, current POS tag, and the shortest dependency path and distance to an opinion word. The features are largely the same as in the **CRF** method.
- **FEMA**: This is another recently proposed domain adaptation method based on cross-domain feature embeddings (Yang and Eisenstein 2015). We use the same feature templates as those for mDA.
- **Direct-1**, **Direct-2**: These two methods naively use the labeled training data from the source domain to train an LSTM model and apply it to the target domain test data. **Direct-2** uses two layers of LSTM. This is to compare with our hierarchical model, which also uses two layers of LSTM.
- **Aux**: This is an unsupervised method where we directly use the rules presented earlier to extract opinion targets.
- **Direct-Aux**: This is a naive way of using both the labeled training data from the source domain and the general syntactic rules. Essentially, we train an LSTM model using the combination of the source domain data with the true labels and the target domain data with the auxiliary labels generated by the rules.

Meanwhile, we have the following variants of our proposed models:

- **Con-Sep** The concatenated network trained with separate training.
- **Con-Joint** The concatenated network trained with joint training.
- **Hier-Sep** The hierarchical network trained with separate training. After the parameters Θ' are learned and the word embeddings updated, they are kept fixed. The subsequent

[1] https://www.cs.uic.edu/∼liub/FBS/sentiment-analysis.html#lexicon

[2] https://code.google.com/archive/p/word2vec/

training only updates Θ in Eqn. (3) and $\mathbf{W}$ and $\mathbf{b}$ in Eqn. (4).

- **Hier-Joint** The hierarchical network trained with joint training.

Experiment Results

We show the $F1$ scores of the various methods under different source-target domain settings in Table 3. From the table we can observe the following: (1) Our proposed model with the hierarchical network and joint learning can outperform all the other methods under all settings except one. Furthermore, most of the time the improvement is statistically significant. This shows the advantage of our proposed model with the Hier-Joint setting. (2) Our proposed model with the other settings (Con-Sep, Con-Joint and Hier-Sep) also tend to work well in many cases, outperforming the baselines. This shows that in general our idea of learning a hidden representation using the auxiliary labels is effective. (3) Comparison among Con-Sep, Con-Joint, Hier-Sep and Hier-Joint shows that using joint learning helps and using the hierarchical network generally is better than using the concatenated network. (4) Among the baselines, Direct-2 tends to work well in general except when Restaurant is the target domain, in which case Aux and Direct+Aux generally work better. This shows that none of the baselines we consider is guaranteed to work well in a cross-domain setting for the opinion target extraction task.

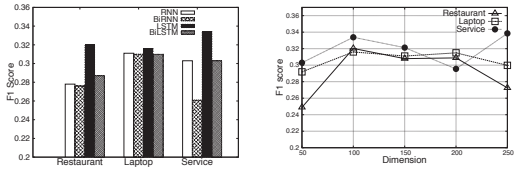
Overall, the results demonstrate that our hierarchical network with joint learning can integrate labeled dataset from the source domain with domain-independent syntactic rules well. The reinforcement between these two types of information makes this model more effective than other models for cross-domain opinion target extraction.

To understand how our Hier-Joint model obtains better performance over the others, we compare the precision and recall of Hier-Joint with all the other baselines. We find that Hier-Joint can get both better precision and better recall most of the time. It demonstrates that our hierarchical network with auxiliary labels can discover more targets without bringing in many false positive predictions.

Analysis

Comparison of different RNNs The results above are based on LSTM, which is a special case of RNN. To compare the effectiveness of different types of RNNs, we pick the Device dataset as the target domain and compare the performance of different RNNs. We consider standard RNN, bi-directional RNN, LSTM and bi-directional LSTM. The results are shown in Figure 3(a) with X-axis showing the source domain. We can see that LSTM consistently achieves better $F1$ scores than the other three RNN models. Its advantage is the most obvious when we use Restaurant or Service as the source domain.

Effect of the hidden layer dimension We also study the effect of using different hidden layer dimensions. We show the results in Figure 3(b). The results are from the Hier-Joint model, which is the best among our proposed models. We again use Device as the target domain. We can see that the $F1$ score first goes up. It then reaches the optimal value at dimension 100 and then starts to go down. The likely reason for the drop in $F1$ score is that the model starts to overfit the training data when the hidden layer dimension becomes too high.



(a) RNNs (b) Hidden dimensions

Figure 3: Effect of different RNNs and hidden dimensions.

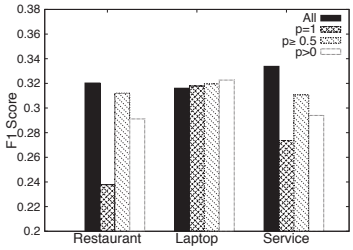


Figure 4: F1 scores of Hier-Joint under different rule selection threshold. p is the precision of the selected rules on the source domain data.

Rule Pruning Using different rules to generate auxiliary labels may lead to different results. To evaluate the effect of different rules at a finer-grained level, we instantiate all general rules listed in Table 1 with opinion words and get 4×6790 concrete rules, where 6790 is the number of opinion words we use. We then calculate the precision of each rule on the source domain and use a threshold to select them. The selected concrete rules are then used to generate auxiliary labels. We test three different thresholds and show the results when using Device as the target domain in Figure 4. We can see that only when Laptop is the source domain, we can get better performance by selecting a subset of the rules. The likely reason is that the Laptop data is similar to the Device data, so pruning rules on the source domain can benefit the extraction task on the target domain. However, when the source domain is not very similar to the target domain, it is better to use all rules without pruning.

Conclusion

In this work, we propose two RNN-based neural networks for cross-domain opinion target extraction. We first use unsupervised syntactic rules to generate an auxiliary label sequence for each sentence. We then train our models using both the true labels and the auxiliary labels. By leveraging knowledge from labeled training data and domain-independent syntactic rules at the same time, our Hierarchical Network with joint learning can learn a robust vector

Data	CRF	mDA	FEMA	Direct-1	Direct-2	Aux	Direct-Aux	Con-Sep	Con-Joint	Hier-Sep	Hier-Joint
L-R	0.170	0.243	0.350	0.301	0.364	0.390	0.436	0.302	0.338	0.436	**0.467**†
D-R	0.025	0.213	0.207	0.306	0.352	0.390	0.432	0.280	0.281	0.451†	**0.504**†
S-R	0.170	0.325	0.376	0.439	0.458	0.390	0.434	0.458	0.421	0.479†	**0.520**†
R-L	0.109	0.209	0.266	0.277	0.290	0.199	0.210	0.248	0.292	0.269	**0.317**†
D-L	0.245	0.257	0.268	0.323	0.353	0.199	0.197	0.334	0.329	0.252	**0.362**†
S-L	0.116	0.146	0.150	0.252	0.249	0.199	0.219	0.256	0.288†	0.260	**0.300**†
R-D	0.090	0.172	0.229	0.246	0.225	0.233	0.246	0.269†	0.295†	0.300†	**0.320**†
L-D	0.270	0.294	0.296	0.296	0.300	0.233	0.226	0.308	0.316	0.279	**0.316**
S-D	0.097	0.169	0.187	0.283	0.275	0.233	0.241	0.243	0.283	0.309†	**0.334**†
R-S	0.088	0.131	0.108	0.146	0.175	0.151	0.127	0.167	0.145	0.180†	**0.198**†
L-S	0.086	0.131	0.148	0.152	0.183	0.151	0.126	0.150	0.185†	0.111	**0.234**†
D-S	0.045	0.095	0.088	0.165	0.151	0.151	0.132	0.166	0.186†	**0.241**†	0.235†
Average	0.126	0.199	0.223	0.265	0.281	0.243	0.252	0.265	0.280	0.297	**0.342**

Table 3: F1 scores achieved by the various methods we consider. The *Data* column shows the source and the target domains, where *L* stands for laptop, *R* stands for restaurant, *D* stands for device and *S* stands for service. † indicates that the result is statistically significantly better than CRF, mDA, FEMA, Direct-1, Direct-2, Aux and Direct-Aux with $p < 0.01$ based on McNemar's test. As an upper bound, we note that the F1 scores on the four domains *R*, *L*, *D* and *S* when trained on in-domain data are 0.779, 0.766, 0.451 and 0.438, respectively.

representation that is useful across domains and outperform several strong baselines. This work shows that it is a promising direction to boost RNNs with rules and auxiliary tasks for opinion target extraction.

Acknowledgement

This research is supported by the National Research Foundation, Prime Minister's Office, Singapore under its International Research Centres in Singapore Funding Initiative.

References

Blitzer, J.; Dredze, M.; and Pereira, F. 2007. Biographies, bollywood, boom-boxes and blenders: Domain adaptation for sentiment classification. In *ACL*, 440–447.

Blitzer, J.; McDonald, R.; and Pereira, F. 2006. Domain adaptation with structural correspondence learning. In *EMNLP*, 120–128.

Chen, M.; Xu, Z. E.; Weinberger, K. Q.; and Sha, F. 2012. Marginalized denoising autoencoders for domain adaptation. In *ICML*, 767–774.

Chernyshevich, M. 2014. IHS R&D Belarus: Cross-domain extraction of product features using conditional random fields. In *SemEval*, 309.

Hochreiter, S., and Schmidhuber, J. 1997. Long short-term memory. *Neural Comput.* 1735–1780.

Hu, M., and Liu, B. 2004. Mining and summarizing customer reviews. In *KDD*, 168–177.

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

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

Kessler, J. S.; Eckert, M.; Clark, L.; and Nicolov, N. 2010. The 2010 icwsm jdpa sentment corpus for the automotive domain. In *ICWSM workshop*.

Lafferty, J. D.; McCallum, A.; and Pereira, F. C. N. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 282–289.

Liu, P.; Joty, S.; and Meng, H. 2015. Fine-grained opinion mining with recurrent neural networks and word embeddings. In *EMNLP*, 1433–1443.

Pan, S. J., and Yang, Q. 2010. A survey on transfer learning. *IEEE TKDE* 22(10):1345–1359.

Pontiki, M.; Galanis, D.; Pavlopoulos, J.; Papageorgiou, H.; Androutsopoulos, I.; and Manandhar, S. 2014. Semeval-2014 task 4: Aspect based sentiment analysis. In *SemEval*, 27–35.

Pontiki, M.; Galanis, D.; Papageorgiou, H.; Manandhar, S.; and Androutsopoulos, I. 2015. Semeval-2015 task 12: Aspect based sentiment analysis. In *SemEval*, 486–495.

Qiu, G.; Liu, B.; Bu, J.; and Chen, C. 2011. Opinion word expansion and target extraction through double propagation. *Computational Linguistics* 37:9–27.

Schouten, K., and Frasincar, F. 2016. Survey on aspect-level sentiment analysis. *IEEE TKDE* 813–830.

Toprak, C.; Jakob, N.; and Gurevych, I. 2010. Sentence and expression level annotation of opinions in user-generated discourse. In *ACL*, 575–584.

Wang, W.; Pan, S. J.; Dahlmeir, D.; and Xiao, X. 2016. Recursive neural conditional random fields for aspect-based sentiment analysis. In *EMNLP*, 616–626.

Yang, B., and Cardie, C. 2013. Joint inference for fine-grained opinion extraction. In *ACL*, 1640–1649.

Yang, Y., and Eisenstein, J. 2015. Unsupervised multi-domain adaptation with feature embeddings. In *NAACL*, 672–682.

Yin, Y.; Wei, F.; Dong, L.; Xu, K.; Zhang, M.; and Zhou, M. 2016. Unsupervised word and dependency path embeddings for aspect term extraction. In *IJCAI*, 2979–2985.

Zhang, M.; Zhang, Y.; and Vo, D. 2015. Neural networks for open domain targeted sentiment. In *EMNLP*, 612–621.

Zhuang, L.; Jing, F.; and Zhu, X.-Y. 2006. Movie review mining and summarization. In *CIKM*, 43–50.